

# Distributed Stochastic Optimization via Correlated Scheduling

Michael J. Neely

University of Southern California

<http://www-bcf.usc.edu/~mjneely>

**Abstract**—This paper considers a problem where multiple users make repeated decisions based on their own observed events. The events and decisions at each time step determine the values of a utility function and a collection of penalty functions. The goal is to make distributed decisions over time to maximize time average utility subject to time average constraints on the penalties. An example is a collection of power constrained sensor nodes that repeatedly report their own observations to a fusion center. Maximum time average utility is fundamentally reduced because users do not know the events observed by others. Optimality is characterized for this distributed context. It is shown that optimality is achieved by correlating user decisions through a commonly known pseudorandom sequence. An optimal algorithm is developed that chooses pure strategies at each time step based on a set of time-varying weights.

## I. INTRODUCTION

Consider a multi-user system that operates over discrete time with unit time slots  $t \in \{0, 1, 2, \dots\}$ . There are  $N$  users. At each time slot  $t$ , each user  $i$  observes a *random event*  $\omega_i(t)$  and makes a *control action*  $\alpha_i(t)$  based on this observation. Let  $\omega(t)$  and  $\alpha(t)$  be vectors of these values:

$$\begin{aligned}\omega(t) &= (\omega_1(t), \omega_2(t), \dots, \omega_N(t)) \\ \alpha(t) &= (\alpha_1(t), \alpha_2(t), \dots, \alpha_N(t))\end{aligned}$$

For each slot  $t$ , these vectors determine the values of a *system utility*  $u(t)$  and a collection of *system penalties*  $p_1(t), \dots, p_K(t)$  (for some non-negative integer  $K$ ) via real-valued functions:

$$\begin{aligned}u(t) &= \hat{u}(\alpha(t), \omega(t)) \\ p_k(t) &= \hat{p}_k(\alpha(t), \omega(t)) \quad \forall k \in \{1, \dots, K\}\end{aligned}$$

The functions  $\hat{u}(\cdot)$  and  $\hat{p}_k(\cdot)$  are arbitrary and can possibly be negative. Negative penalties can be used to represent desirable *system rewards*.

The goal is to make distributed decisions over time that maximize time average utility subject to time average constraints on the penalties. Central to this problem is the assumption that each user  $i$  can only observe  $\omega_i(t)$ , and cannot observe the value of  $\omega_j(t)$  for other users  $j \neq i$ . Further, each user  $i$  only knows its own action  $\alpha_i(t)$ , but does not know the actions  $\alpha_j(t)$  of others. Therefore, each user only knows a portion of the arguments that go into the

functions  $\hat{u}(\alpha(t), \omega(t))$  and  $\hat{p}_k(\alpha(t), \omega(t))$  for each slot  $t$ . This uncertainty fundamentally restricts the time averages that can be achieved.

Specifically, assume the random event vector  $\omega(t)$  is independent and identically distributed (i.i.d.) over slots (possibly correlated over entries in each slot). The vector  $\omega(t)$  takes values in some abstract *event space*  $\Omega = \Omega_1 \times \Omega_2 \times \dots \times \Omega_N$ , where  $\omega_i(t) \in \Omega_i$  for all  $i \in \{1, \dots, N\}$  and all slots  $t$ . Similarly, assume  $\alpha(t)$  is chosen in some abstract *action space*  $\mathcal{A} = \mathcal{A}_1 \times \mathcal{A}_2 \times \dots \times \mathcal{A}_N$ , where  $\alpha_i(t) \in \mathcal{A}_i$  for all  $i \in \{1, \dots, N\}$  and all slots  $t$ . Let  $\bar{u}$  and  $\bar{p}_k$  be the time average expected utility and penalty incurred by a particular algorithm:<sup>1</sup>

$$\begin{aligned}\bar{u} &= \lim_{t \rightarrow \infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}[u(\tau)] \\ \bar{p}_k &= \lim_{t \rightarrow \infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}[p_k(\tau)]\end{aligned}$$

The following problem is considered:

$$\text{Maximize:} \quad \bar{u} \quad (1)$$

$$\text{Subject to:} \quad \bar{p}_k \leq c_k \quad \forall k \in \{1, \dots, K\} \quad (2)$$

$$\text{Decisions are distributed} \quad (3)$$

where  $c_k$  are a given collection of real numbers that specify constraints on the time average penalties.

The constraint that decisions must be distributed, specified in (3), is not mathematically precise. This constraint is more carefully posed in Section III. Without the distributed scheduling constraint, the problem (1)-(2) reduces to a standard problem of stochastic network optimization and can be solved via the *drift-plus-penalty method* [1]. Such a centralized approach would allow users to coordinate to form an action vector  $\alpha(t)$  based on full knowledge of the event vector  $\omega(t)$ . The time average utility achieved by the best centralized algorithm can be strictly larger than that of the best distributed algorithm. This is shown for an example sensor network problem in Section II.

### A. Applications to sensor networks

The above formulation is useful for a variety of stochastic network optimization problems where distributed agents make their own decisions based on partial system knowledge. An important example is a network of wireless sensor nodes that

<sup>1</sup>For simplicity, it is temporarily assumed that the time averages exist. A more precise formulation is specified in Section III using  $\liminf$  and  $\limsup$ .

The author is with the Electrical Engineering department at the University of Southern California, Los Angeles, CA.

This work is supported in part by one or more of: the NSF Career grant CCF-0747525, NSF grant 1049541, the Network Science Collaborative Technology Alliance sponsored by the U.S. Army Research Laboratory W911NF-09-2-0053.

repeatedly send reports about system events to a fusion center. The goal is to make distributed decisions that maximize time average *quality of information*. This scenario was previously considered by Liu et al. in [2]. There, sensors can provide reports every slot  $t$  using one of multiple *reporting formats*, such as text, image, or video. Sensors can also choose to remain idle on slot  $t$ . Thus, the action spaces  $\mathcal{A}_i$  are the same for all sensors  $i$ :

$$\alpha_i(t) \in \mathcal{A}_i \triangleq \{\text{idle}, \text{text}, \text{image}, \text{video}\} \quad \forall i \in \{1, \dots, N\}$$

where the notation “ $\triangleq$ ” represents *defined to be equal to*. Each format requires a different amount of power and provides a different level of quality. For example, define  $p_i(t)$  as the power incurred by sensor  $i$  on slot  $t$ , where:

$$p_i(t) = \begin{cases} 0 & \text{if } \alpha_i(t) = \text{idle} \\ p_{\text{text}} & \text{if } \alpha_i(t) = \text{text} \\ p_{\text{image}} & \text{if } \alpha_i(t) = \text{image} \\ p_{\text{video}} & \text{if } \alpha_i(t) = \text{video} \end{cases}$$

where  $p_{\text{text}}$ ,  $p_{\text{image}}$ ,  $p_{\text{video}}$  represent powers required for each of the three reporting formats and satisfy:

$$0 < p_{\text{text}} < p_{\text{image}} < p_{\text{video}}$$

Assume that  $\omega_i(t)$  represents the *quality* that sensor  $i$  would bring to the fusion center if it reports the event it observes on slot  $t$  using the video format. Define  $f(\alpha_i(t))$  as the fraction of this quality that is achieved under format  $\alpha_i(t)$ :

$$f(\alpha_i(t)) = \begin{cases} 0 & \text{if } \alpha_i(t) = \text{idle} \\ f_{\text{text}} & \text{if } \alpha_i(t) = \text{text} \\ f_{\text{image}} & \text{if } \alpha_i(t) = \text{image} \\ 1 & \text{if } \alpha_i(t) = \text{video} \end{cases}$$

where

$$0 < f_{\text{text}} < f_{\text{image}} < 1$$

The prior work [2] considers the problem of maximizing time average utility subject to a time average power constraint:

$$\sum_{i=1}^N \bar{p}_i \leq c$$

where  $c$  is some given positive number. Further, that work restricts to the special case when the utility function is a *separable sum* of functions of user  $i$  variables, such as:

$$u(t) = \sum_{i=1}^N f_i(\alpha_i(t)) \omega_i(t)$$

Such separable utilities cannot model the realistic scenario of *information saturation*, where, once a certain amount of utility is achieved on slot  $t$ , there is little value of having additional sensors spend power to deliver additional information on that slot. The current paper considers the case of *arbitrary, possibly non-separable* utility functions. An example is:

$$u(t) = \min \left[ \sum_{i=1}^N f(\alpha_i(t)) \omega_i(t), 1 \right]$$

This means that once a total quality of 1 is accumulated from one or more sensors on slot  $t$ , there is no advantage in having

other sensors report information on that slot. This scenario is significantly more challenging to solve in a distributed context. For example, suppose the  $\omega_i(t)$  variables are binary valued, representing whether or not sensor  $i$  observes an event on slot  $t$ . Suppose  $\omega_1(t) = \omega_2(t) = 1$ . Utility is maximized if either sensor 1 or sensor 2 decides to report in the video format. Power is wasted if they both send video reports. However, sensor 1 does not know the value of  $\omega_2(t)$ , sensor 2 does not know the value of  $\omega_1(t)$ , and neither sensor knows what format will be selected by the other.

## B. Applications to wireless multiple access

The general formulation of this paper can also treat simple forms of distributed multiple access problems. Again suppose there are  $N$  wireless sensors that report to a fusion center. For each  $i \in \{1, \dots, N\}$ , define  $\omega_i(t)$  as the *quality* that a transmission from sensor  $i$  would bring to the system if it transmits on slot  $t$ . Define  $\alpha_i(t)$  as a binary value that is 1 if sensor  $i$  transmits on slot  $t$ , and 0 else. Assume the network operates according to a simple collision model, where a transmission from sensor  $i$  is successful on slot  $t$  if and only if it is the only sensor that transmits on that slot:

$$u(t) = \sum_{i=1}^N \omega_i(t) \left[ \alpha_i(t) \prod_{j \neq i} (1 - \alpha_j(t)) \right] \quad (4)$$

The above utility function is non-separable. Concurrent work in [3] considers a similar utility function for wireless energy harvesting applications.

## C. Contributions and related work

The framework of partial knowledge at each user is similar in spirit to a *multi-player Bayesian game* [4][5]. There, the goal is to design competitive strategies that lead to a Nash equilibrium. This is significantly different from the goal of the current paper. The current paper is not concerned with competition or equilibrium. Rather, there is a single utility function that all users desire to maximize. Distributed algorithms are developed to maximize time average utility subject to time average penalty constraints.

This paper shows that an optimal distributed algorithm can be designed by having users correlate their decisions through an independent source of common randomness (Section III). Related notions of commonly shared randomness are used in game theory to define a *correlated equilibrium*, which is typically easier to compute than a standard Nash equilibrium [6][7][5][4]. For the current paper, the shared randomness is crucial for solving the distributed optimization problem. This paper shows that optimality can be achieved by using a shared random variable with  $K + 1$  possible outcomes, where  $K$  is the number of penalty constraints. The solution is computable through a linear program. Unfortunately, the linear program can have a very large number of variables, even for 2-user problems. A reduction to polynomial complexity is shown to be possible in certain cases (Section IV). This paper also develops an online algorithm that chooses pure strategies every slot based on a set of weights that are updated at the end

of each slot (Section V). The online technique is based on Lyapunov optimization concepts [1][8][9].

Much prior work on network optimization treats scenarios where it is possible to find distributed solutions with no loss of optimality. For example, network flow problems that are described by linear or separable convex programs can be optimally solved in a distributed manner [10][11][12][9]. Problems where network nodes want to average sensor data [13] or compute convex programs [14] have distributed solutions. Work in [15] solves for an optimal vector of parameters associated with an infinite horizon Markov decision problem using distributed agents. Work in [16][17][18] develops distributed multiple access methods that converge to optimality. However, the above problems do not have random events that create a fundamental gap between centralized and distributed performance.

Recent work in [19] derives structural results for distributed optimization in Markov decision systems with delayed information. Such problems *do* exhibit gaps between centralized and distributed scheduling. The use of *private information* in [19] is similar in spirit to the assumption in the current paper that each user observes its own random event  $\omega_i(t)$ . The work [19] derives a sufficient statistic for dynamic programming. It does not consider time average constraints and its solutions do not involve correlated scheduling via a pseudorandom sequence. Recent work in [3] considers distributed reporting of events with different qualities, but considers a more restrictive class of policies that do not use correlated scheduling. The current paper treats a different model than [19] and [3], and shows that correlated scheduling is necessary in systems with constraints. Further, the current paper provides complexity reduction results under a preferred action property (Section IV) and provides an online algorithm that does not require a-priori knowledge of event probabilities (Section V).

## II. EXAMPLE SENSOR NETWORK PROBLEM

This section illustrates the benefits of using a common source of randomness for a simple example network. Suppose the network has two sensors that operate over time slots  $t \in \{0, 1, 2, \dots\}$ . Every slot, the sensors observe the state of a particular system and choose whether or not to report their observations to a fusion center. Let  $\omega_i(t)$  be a binary variable that is 1 if sensor  $i$  observes an event on slot  $t$ , and 0 else. Let  $\alpha_1(t)$  and  $\alpha_2(t)$  be the slot  $t$  decision variables, so that  $\alpha_i(t) = 1$  if sensor  $i$  reports on slot  $t$ , and  $\alpha_i(t) = 0$  otherwise. Suppose the fusion center trusts sensor 1 more than sensor 2. The utility  $u(t)$  is:

$$u(t) = \min[\omega_1(t)\alpha_1(t) + \omega_2(t)\alpha_2(t)/2, 1]$$

so that the deterministic function  $\hat{u}(\cdot)$  is given by:

$$\hat{u}(\alpha_1, \alpha_2, \omega_1, \omega_2) = \min[\omega_1\alpha_1 + \omega_2\alpha_2/2, 1] \quad (5)$$

Therefore,  $u(t) \in \{0, 1/2, 1\}$  for all slots  $t$ . If  $\omega_1(t) = 1$  and sensor 1 reports on slot  $t$ , there is no utility increase if sensor 2 also reports.

Each report uses one unit of power. Let  $p_i(t)$  be the power incurred by sensor  $i$  on slot  $t$ , being 1 if it reports

its observation, and 0 otherwise. The power penalties for  $i \in \{1, 2\}$  are:

$$p_i(t) = \alpha_i(t) \quad (6)$$

so that  $\hat{p}_i(\alpha_1, \alpha_2, \omega_1, \omega_2) = \alpha_i$  for  $i \in \{1, 2\}$ . Each sensor  $i$  can choose *not* to report an observation in order to save power. The difficulty is that neither sensor knows what event was observed by the other. Therefore, a distributed algorithm might send reports from *both* sensors on a given slot. A centralized scheduler would avoid this because it wastes power without increasing utility.

Suppose that  $\omega_1(t)$  and  $\omega_2(t)$  are independent of each other and i.i.d. over slots, with:

$$Pr[\omega_1(t) = 1] = 3/4, \quad Pr[\omega_1(t) = 0] = 1/4$$

$$Pr[\omega_2(t) = 1] = 1/2, \quad Pr[\omega_2(t) = 0] = 1/2$$

To fix a specific numerical example, consider the following problem:

$$\text{Maximize:} \quad \bar{u} \quad (7)$$

$$\text{Subject to:} \quad \bar{p}_1 \leq 1/3, \quad \bar{p}_2 \leq 1/3 \quad (8)$$

$$\text{Decisions are distributed} \quad (9)$$

### A. Independent reporting

Consider the following class of *independent scheduling* algorithms: Each sensor  $i$  independently decides to report with probability  $\theta_i$  if it observes  $\omega_i(t) = 1$  (it does not report if  $\omega_i(t) = 0$ ). Since  $\omega(t)$  is i.i.d. over slots, the resulting sequences  $\{u(t)\}_{t=0}^\infty$ ,  $\{p_1(t)\}_{t=0}^\infty$ ,  $\{p_2(t)\}_{t=0}^\infty$  are i.i.d. over slots. The time averages are:

$$\bar{p}_1 = \frac{3}{4}\theta_1, \quad \bar{p}_2 = \frac{1}{2}\theta_2$$

$$\begin{aligned} \bar{u} &= \mathbb{E}[u(t)|\omega_1(t) = 1, \omega_2(t) = 0] \frac{3}{4} \frac{1}{2} \\ &+ \mathbb{E}[u(t)|\omega_1(t) = 0, \omega_2(t) = 1] \frac{1}{4} \frac{1}{2} \\ &+ \mathbb{E}[u(t)|\omega_1(t) = \omega_2(t) = 1] \frac{3}{4} \frac{1}{2} \\ &= \frac{3}{4} \frac{1}{2} \theta_1 + \frac{1}{4} \frac{1}{2} (\theta_2/2) + \frac{3}{4} \frac{1}{2} (\theta_1 + (1 - \theta_1)\theta_2/2) \end{aligned}$$

For this class of algorithms, utility is maximized by choosing  $\theta_1$  and  $\theta_2$  to meet the power constraints with equality. This leads to  $\theta_1 = 4/9$ ,  $\theta_2 = 2/3$ . The resulting utility is:

$$\bar{u} = 4/9 \approx 0.44444$$

### B. Correlated reporting

As an alternative, consider the following three strategies:

- Strategy 1:  $\omega_1(t) = 1 \implies \alpha_1(t) = 1$  (else,  $\alpha_1(t) = 0$ ). Sensor 2 always chooses  $\alpha_2(t) = 0$ .
- Strategy 2:  $\omega_2(t) = 1 \implies \alpha_2(t) = 1$  (else,  $\alpha_2(t) = 0$ ). Sensor 1 always chooses  $\alpha_1(t) = 0$ .
- Strategy 3:  $\omega_1(t) = 1 \implies \alpha_1(t) = 1$  (else,  $\alpha_1(t) = 0$ ).  $\omega_2(t) = 1 \implies \alpha_2(t) = 1$  (else,  $\alpha_2(t) = 0$ ).

The above three strategies are *pure strategies* because  $\alpha_i(t)$  is a deterministic function of  $\omega_i(t)$  for each sensor  $i$ . Now let

$X(t)$  be an external source of randomness that is commonly known at both sensors on slot  $t$ . Assume  $X(t)$  is independent of everything else in the system, and is i.i.d. over slots with:

$$\begin{aligned} \Pr[X(t) = 1] &= \theta_1 \\ \Pr[X(t) = 2] &= \theta_2 \\ \Pr[X(t) = 3] &= \theta_3 \end{aligned}$$

where  $\theta_1, \theta_2, \theta_3$  are probabilities that sum to 1. Consider the following algorithm: On slot  $t$ , if  $X(t) = m$  then choose strategy  $m$ , where  $m \in \{1, 2, 3\}$ . This algorithm can be implemented by letting  $X(t)$  be a pseudorandom sequence that is installed in both sensors at time 0. The resulting time averages are:

$$\begin{aligned} \bar{p}_1 &= (\theta_1 + \theta_3)\frac{3}{4}, \quad \bar{p}_2 = (\theta_2 + \theta_3)\frac{1}{2} \\ \bar{u} &= \theta_1\frac{3}{4} + \theta_2\frac{1}{2} + \theta_3(\frac{3}{4} + \frac{1}{4}\frac{1}{2}) \end{aligned}$$

A simple linear program can be used to compute the optimal  $\theta_1, \theta_2, \theta_3$  probabilities for this algorithm structure. The result is  $\theta_1 = 1/3, \theta_2 = 5/9, \theta_3 = 1/9$ . The resulting time average utility is:

$$\bar{u} = 23/48 \approx 0.47917$$

This is strictly larger than the time average utility of 0.44444 achieved by the independent reporting algorithm. Thus, performance can be strictly improved by correlating reports via a common source of randomness. Alternatively, the same time averages can be achieved by *time sharing*: The two sensors agree to use a periodic schedule of period 9 slots. The first 3 slots of the period use strategy 1, the next 5 slots use strategy 2, and the final slot uses strategy 3.

### C. Centralized reporting

Suppose sensors coordinate by observing  $(\omega_1(t), \omega_2(t))$  and then cooperatively selecting  $(\alpha_1(t), \alpha_2(t))$ . It turns out that an optimal centralized policy is as follows [1]: Every slot  $t$ , observe  $(\omega_1(t), \omega_2(t))$  and choose  $(\alpha_1(t), \alpha_2(t))$  as follows:

- $(\omega_1(t), \omega_2(t)) = (0, 0) \implies (\alpha_1(t), \alpha_2(t)) = (0, 0)$ .
- $(\omega_1(t), \omega_2(t)) = (0, 1) \implies (\alpha_1(t), \alpha_2(t)) = (0, 1)$ .
- If  $(\omega_1(t), \omega_2(t)) = (1, 0)$ , independently choose:

$$(\alpha_1(t), \alpha_2(t)) = \begin{cases} (1, 0) & \text{with probability } 8/9 \\ (0, 0) & \text{with probability } 1/9 \end{cases}$$

- If  $(\omega_1(t), \omega_2(t)) = (1, 1)$ , independently choose:

$$(\alpha_1(t), \alpha_2(t)) = \begin{cases} (0, 1) & \text{with probability } 5/9 \\ (0, 0) & \text{with probability } 4/9 \end{cases}$$

The resulting optimal centralized time average utility is:

$$\bar{u} = 0.5$$

This is larger than the value 0.47917 achieved by the distributed algorithm of the previous subsection.

The question remains: Is it possible to construct some other distributed algorithm that yields  $\bar{u} > 0.47917$ ? Results in the next section imply this is impossible. Thus, the correlated reporting algorithm of the previous subsection optimizes time average utility over all possible distributed algorithms that satisfy the constraints. Therefore, for this example, there is a *fundamental gap* between the performance of the best centralized algorithm and the best distributed algorithm.

## III. CHARACTERIZING OPTIMALITY

This section considers the general  $N$  user problem and characterizes optimality over all possible distributed algorithms. Recall that:

$$\begin{aligned} \omega(t) &\in \Omega = \Omega_1 \times \cdots \times \Omega_N \\ \alpha(t) &\in \mathcal{A} = \mathcal{A}_1 \times \cdots \times \mathcal{A}_N \end{aligned}$$

where the vectors  $\omega(t)$  are i.i.d. over slots (possibly correlated over entries in each slot). Assume that the sets  $\Omega_i$  and  $\mathcal{A}_i$  are finite with sizes denoted  $|\Omega_i|$  and  $|\mathcal{A}_i|$ . For each  $\omega \in \Omega$  define:

$$\pi(\omega) = \Pr[\omega(t) = \omega]$$

Define the *history*  $\mathcal{H}(t)$  by:

$$\mathcal{H}(t) \triangleq \{(\omega(0), \alpha(0)), \dots, (\omega(t-1), \alpha(t-1))\}$$

This section considers all distributed algorithms, including those where all users know the full history  $\mathcal{H}(t)$ . Such information might be available through a feedback message that specifies  $(\alpha(t), \omega(t))$  at the end of each slot  $t$ . Theorem 1 shows that optimality can be achieved *without* this history information.

First, it is important to make the distributed scheduling constraint (3) mathematically precise. One might attempt to use the following condition. For all slots  $t$ , the decisions made by each user  $i \in \{1, \dots, N\}$  must satisfy:

$$\begin{aligned} \Pr[\alpha_i(t) = \alpha_i | \omega_i(t) = \omega_i, \mathcal{H}(t)] \\ = \Pr[\alpha_i(t) = \alpha_i | \omega(t) = \omega, \mathcal{H}(t)] \end{aligned} \quad (10)$$

for all vectors  $\omega = (\omega_1, \dots, \omega_N) \in \Omega_1 \times \cdots \times \Omega_N$  and all  $\alpha_i \in \mathcal{A}_i$ . The condition (10) specifies that  $\alpha_i(t)$  is conditionally independent of  $(\omega_j(t))_{j \neq i}$  given  $\omega_i(t)$ ,  $\mathcal{H}(t)$ . While this condition is indeed required, it turns out that it is not restrictive enough. Appendix B provides an example utility function for which there is an algorithm that satisfies (10) but yields expected utility strictly larger than that of any “true” distributed algorithm (as defined in the next subsection).

### A. The distributed scheduling constraint

An algorithm for selecting  $\alpha(t)$  over slots  $t \in \{0, 1, 2, \dots\}$  is *distributed* if:

- There is an abstract set  $\mathcal{X}$ , called a *common information set*.
- There is a sequence of *commonly known random elements*  $X(t) \in \mathcal{X}$  such that  $\omega(t)$  is independent of  $X(t)$  for each  $t \in \{0, 1, 2, \dots\}$ .
- There are deterministic functions  $f_i(\omega_i, X)$  for each  $i \in \{1, \dots, N\}$  of the form:

$$f_i : \Omega_i \times \mathcal{X} \rightarrow \mathcal{A}_i$$

- The decisions  $\alpha_i(t)$  satisfy the following for all slots  $t$ :

$$\alpha_i(t) = f_i(\omega_i(t), X(t)) \text{ for all } i \in \{1, \dots, N\} \quad (11)$$

The above definition includes a wide class of algorithms. Intuitively, the random elements  $X(t)$  can be designed as any source of common randomness on which users can base their

decisions. For example,  $X(t)$  can be designed to have the form:

$$X(t) = (t, \mathcal{H}(t), Y(t))$$

where  $Y(t)$  is a random element with support and distribution that can possibly depend on  $\mathcal{H}(t)$  as well as past values  $Y(\tau)$  for  $\tau < t$ . The only restriction is that  $X(t)$  is independent of  $\omega(t)$ . Because the  $\omega(t)$  vectors are i.i.d. over slots,  $X(t)$  can be based on any events that occur before slot  $t$ .

### B. The optimization problem

For notational convenience, define:

$$\begin{aligned} p_0(t) &\triangleq -u(t) \\ \hat{p}_0(\alpha(t), \omega(t)) &\triangleq -\hat{u}(\alpha(t), \omega(t)) \end{aligned}$$

Maximizing the time average expectation of  $u(t)$  is equivalent to minimizing the time average expectation of  $p_0(t)$ . For each  $k \in \{0, 1, \dots, K\}$  and each slot  $t > 0$  define:

$$\bar{p}_k(t) \triangleq \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}[p_k(\tau)]$$

The goal is to design a distributed algorithm that solves the following:

$$\text{Minimize:} \quad \limsup_{t \rightarrow \infty} \bar{p}_0(t) \quad (12)$$

$$\text{Subject to:} \quad \limsup_{t \rightarrow \infty} \bar{p}_k(t) \leq c_k \quad \forall k \in \{1, \dots, K\} \quad (13)$$

$$\text{Condition (11) holds } \forall t \in \{0, 1, 2, \dots\} \quad (14)$$

It is assumed throughout this paper that the constraints (13)-(14) are *feasible*. Define  $p_0^{opt}$  as the infimum of all limiting  $\bar{p}_0(t)$  values (12) achievable by algorithms that satisfy the constraints (13)-(14). The infimum is finite because  $p_0(t)$  takes values in the same bounded set for all slots  $t$ .

### C. Optimality via correlated scheduling

A *pure strategy* is defined as a vector-valued function:

$$\mathbf{g}(\omega) = (g_1(\omega_1), g_2(\omega_2), \dots, g_N(\omega_N))$$

where  $g_i(\omega_i) \in \mathcal{A}_i$  for all  $i \in \{1, \dots, N\}$  and all  $\omega_i \in \Omega_i$ . The function  $\mathbf{g}(\omega)$  specifies a distributed decision rule where each user  $i$  chooses  $\alpha_i$  as a deterministic function of  $\omega_i$ . Specifically,  $\alpha_i = g_i(\omega_i)$ . The total number of pure strategy functions  $\mathbf{g}(\omega)$  is  $\prod_{i=1}^N |\mathcal{A}_i|^{\Omega_i}$ . Define  $M$  as this number, and enumerate all these vectors by  $\mathbf{g}^{(m)}(\omega)$  for  $m \in \{1, \dots, M\}$ . For each  $m \in \{1, \dots, M\}$  and  $k \in \{0, 1, \dots, K\}$  define:

$$r_k^{(m)} \triangleq \sum_{\omega \in \Omega} \pi(\omega) \hat{p}_k(\mathbf{g}^{(m)}(\omega), \omega) \quad (15)$$

The value  $r_k^{(m)}$  is the expected value of  $p_k(t)$  given that users implement strategy  $\mathbf{g}^{(m)}(\omega)$  on slot  $t$ .

Consider a randomized algorithm that, every slot  $t$ , independently uses strategy  $\mathbf{g}^{(m)}(\omega)$  with probability  $\theta_m$ . For each

$k \in \{0, 1, \dots, K\}$ , the expected penalty  $\mathbb{E}[p_k(t)]$  under such a strategy is:

$$\begin{aligned} \mathbb{E}[p_k(t)] &= \sum_{m=1}^M \theta_m \mathbb{E}[\hat{p}_k(\mathbf{g}^{(m)}(\omega(t)), \omega(t))] \\ &= \sum_{m=1}^M \theta_m r_k^{(m)} \end{aligned}$$

The following linear program optimizes over the  $\theta_m$  probabilities for this specific algorithm structure:

$$\text{Minimize:} \quad \sum_{m=1}^M \theta_m r_0^{(m)} \quad (16)$$

$$\text{Subject to:} \quad \sum_{m=1}^M \theta_m r_k^{(m)} \leq c_k \quad \forall k \in \{1, \dots, K\} \quad (17)$$

$$\theta_m \geq 0 \quad \forall m \in \{1, \dots, M\} \quad (18)$$

$$\sum_{m=1}^M \theta_m = 1 \quad (19)$$

The objective (16) corresponds to minimizing  $\mathbb{E}[p_0(t)]$ , the constraints (17) ensure  $\mathbb{E}[p_k(t)] \leq c_k$  for  $k \in \{1, \dots, K\}$ , and the constraints (18)-(19) ensure the  $\theta_m$  values form a valid probability mass function. Such a randomized algorithm does not use the history  $\mathcal{H}(t)$ . The next theorem shows this algorithm structure is optimal.

*Theorem 1:* Suppose the problem (12)-(14) is feasible. Then the linear program (16)-(19) is feasible, and the optimal objective value (16) is equal to  $p_0^{opt}$ . Furthermore, there exist probabilities  $(\theta_1, \dots, \theta_M)$  that solve the linear program and satisfy  $\theta_m > 0$  for at most  $K + 1$  values of  $m \in \{1, \dots, M\}$ .

*Proof:* See Appendix A. ■

## IV. REDUCED COMPLEXITY

The linear program (16)-(19) uses variables  $(\theta_1, \theta_2, \dots, \theta_M)$ , where  $M$  is the number of pure strategies:

$$M = \prod_{i=1}^N |\mathcal{A}_i|^{\Omega_i}$$

The 2-user sensor network example from Section II has  $|\mathcal{A}_i| = |\Omega_i| = 2$  for  $i \in \{1, 2\}$ , for a total of  $2^2 = 4$  strategy functions  $g_i(\omega_i)$  for each user—hence a total of  $M = 16$  functions  $\mathbf{g}(\omega)$ . However, for each user  $i$ , the two strategy functions  $g_i(\omega_i)$  that give  $g_i(0) = 1$  can be removed from consideration (as it is useless for user  $i$  to report if it observes no event). Thus, the effective number of strategy functions  $g_i(\omega_i)$  for each user is only two, leaving only four functions  $\mathbf{g}(\omega) = (g_1(\omega_1), g_2(\omega_2))$ . The optimal probabilities for switching between these four is given in Section II-B, where it is seen that only  $K + 1 = 3$  strategies have non-zero probabilities.

For general problems, the value of  $M$  can be very large. The remainder of this section shows that, if certain conditions hold, the set of strategy functions can be pruned to a smaller set without loss of optimality. For example, consider a two-user

problem with binary actions, so that  $|\mathcal{A}_i| = 2$  for  $i \in \{1, 2\}$ . Then:

$$M = 2^{|\Omega_1| + |\Omega_2|}$$

If certain conditions hold, strategies can be restricted to a set of size  $\tilde{M}$ , where:

$$\tilde{M} = (|\Omega_1| + 1)(|\Omega_2| + 1)$$

Thus, an exponentially large set is pruned to a smaller set with polynomial size.

#### A. The preferred action property

Suppose the sets  $\mathcal{A}_i$  and  $\Omega_i$  for each user  $i \in \{1, \dots, N\}$  are given by:

$$\mathcal{A}_i = \{0, 1, \dots, |\mathcal{A}_i| - 1\} \quad (20)$$

$$\Omega_i = \{0, 1, \dots, |\Omega_i| - 1\} \quad (21)$$

For notational convenience, for each  $i \in \{1, \dots, N\}$  let  $[\alpha_{\bar{i}}, \alpha_i]$  denote the  $N$ -dimensional vector  $\alpha = (\alpha_1, \dots, \alpha_N)$ , where  $\alpha_{\bar{i}}$  is the  $(N-1)$ -dimensional vector of  $\alpha_j$  components for  $j \neq i$ . This notation facilitates comparison of two vectors that differ in just one coordinate. Define  $\mathcal{A}_{\bar{i}}$  and  $\Omega_{\bar{i}}$  as the set of all possible  $(N-1)$ -dimensional vectors  $\alpha_{\bar{i}}$  and  $\omega_{\bar{i}}$ , respectively.

**Definition 1:** A penalty function  $\hat{p}(\alpha, \omega)$  has the *preferred action property* if for all  $i \in \{1, \dots, N\}$ , all  $\alpha_{\bar{i}} \in \mathcal{A}_{\bar{i}}$ , and all  $\omega_{\bar{i}} \in \Omega_{\bar{i}}$ , one has:

$$\begin{aligned} & \hat{p}([\alpha_{\bar{i}}, \alpha], [\omega_{\bar{i}}, \omega]) - \hat{p}([\alpha_{\bar{i}}, \beta], [\omega_{\bar{i}}, \omega]) \\ & \geq \hat{p}([\alpha_{\bar{i}}, \alpha], [\omega_{\bar{i}}, \gamma]) - \hat{p}([\alpha_{\bar{i}}, \beta], [\omega_{\bar{i}}, \gamma]) \end{aligned}$$

whenever  $\alpha, \beta$  are values in  $\mathcal{A}_i$  that satisfy  $\alpha > \beta$ , and  $\omega, \gamma$  are values in  $\Omega_i$  that satisfy  $\omega < \gamma$ .

Intuitively, the above definition means that if user  $i$  compares the difference in penalty under the actions  $\alpha_i(t) = \alpha$  and  $\alpha_i(t) = \beta$  (where  $\alpha > \beta$ ), this difference is non-increasing in the user  $i$  observation  $\omega_i(t)$  (assuming all other actions and events  $\alpha_{\bar{i}}$  and  $\omega_{\bar{i}}$  are held fixed).

For example, any function  $\hat{p}(\alpha, \omega)$  that does not depend on  $\omega$  trivially satisfies the preferred action property. This is the case for the  $\hat{p}_1(\cdot)$  and  $\hat{p}_2(\cdot)$  functions in (6) used to represent power expenditures for the sensor network example of Section II. Further, the utility function (5) in that example yields  $\hat{p}_0(\cdot) = -\hat{u}(\cdot)$  that satisfies the preferred action property, as shown by the next lemma.

**Lemma 1:** Suppose  $\mathcal{A}_i = \{0, 1\}$  for  $i \in \{1, \dots, N\}$ ,  $\Omega_i$  is given by (21), and define:

$$\hat{u}(\alpha, \omega) = \min \left[ \sum_{i=1}^N \phi_i(\omega_i) \alpha_i, b \right]$$

for some (real-valued) constant  $b$  and some (real-valued) non-decreasing functions  $\phi_i(\omega_i)$ . Then the penalty function  $\hat{p}_0(\alpha, \omega) = -\hat{u}(\alpha, \omega)$  has the preferred action property.

**Lemma 2:** Suppose  $\mathcal{A}_i = \{0, 1\}$  for  $i \in \{1, \dots, N\}$ ,  $\Omega_i$  is given by (21), and define the utility function  $\hat{u}(\alpha, \omega)$  according to the multi-access example equation (4). Then the penalty function  $\hat{p}_0(\alpha, \omega) = -\hat{u}(\alpha, \omega)$  has the preferred action property.

**Lemma 3:** Suppose  $\mathcal{A}_i$  and  $\Omega_i$  are given by (20)-(21). Define  $\hat{p}(\alpha, \omega)$  by:

$$\hat{p}(\alpha, \omega) = \prod_{i=1}^N \phi_i(\omega_i) \psi_i(\alpha_i)$$

where  $\phi_i(\omega_i)$ ,  $\psi_i(\alpha_i)$  are non-negative functions for all  $i \in \{1, \dots, N\}$ . Suppose that for each  $i \in \{1, \dots, N\}$ ,  $\phi_i(\omega_i)$  is non-increasing in  $\omega_i$  and  $\psi_i(\alpha_i)$  is non-decreasing in  $\alpha_i$ . Then  $\hat{p}(\alpha, \omega)$  has the preferred action property.

**Lemma 4:** Suppose  $\mathcal{A}_i$  and  $\Omega_i$  are given by (20)-(21). Suppose  $\hat{p}_1(\alpha, \omega), \dots, \hat{p}_R(\alpha, \omega)$  are a collection of functions that have the preferred action property (where  $R$  is a given positive integer). Then for any non-negative weights  $w_1, \dots, w_R$ , the following function has the preferred action property:

$$\hat{p}(\alpha, \omega) = \sum_{r=1}^R w_r \hat{p}_r(\alpha, \omega)$$

The proofs of Lemmas 1-4 are given in Appendix C.

#### B. Independent events and reduced complexity

Consider the special case when the components of  $\omega(t) = (\omega_1(t), \dots, \omega_N(t))$  are mutually independent, so that:

$$\pi(\omega) = \prod_{i=1}^N q_i(\omega_i) \quad (22)$$

where:

$$q_i(\omega_i) \triangleq \Pr[\omega_i(t) = \omega_i]$$

Without loss of generality, assume  $q_i(\omega_i) > 0$  for all  $i \in \{1, \dots, N\}$  and all  $\omega_i \in \Omega_i$ . Recall that a pure strategy  $\mathbf{g}(\omega)$  is composed of individual *strategy functions*  $g_i(\omega_i)$  for each user  $i$ :

$$\mathbf{g}(\omega) = (g_1(\omega_1), \dots, g_N(\omega_N))$$

**Theorem 2:** (Non-decreasing strategy functions) Suppose  $\mathcal{A}_i$  and  $\Omega_i$  are given by (20)-(21). If all penalty functions  $\hat{p}_k(\alpha, \omega)$  for  $k \in \{0, 1, \dots, K\}$  have the preferred action property, and if the random event process  $\omega(t)$  satisfies the independence property (22), then it suffices to restrict attention to strategy functions  $g_i(\omega_i)$  that are non-decreasing in  $\omega_i$ .

**Proof:** The proof uses an interchange argument. Fix  $m \in \{1, \dots, M\}$ ,  $i \in \{1, \dots, N\}$ , and fix two elements  $\omega$  and  $\gamma$  in  $\Omega_i$  that satisfy  $\omega < \gamma$ . Suppose the linear program (16)-(19) places weight  $\theta_m > 0$  on a strategy function  $\mathbf{g}^{(m)}(\omega)$  that satisfies  $g_i^{(m)}(\omega) > g_i^{(m)}(\gamma)$  (so the non-decreasing requirement is violated). The goal is to show this can be replaced by new strategies that do not violate the non-decreasing requirement for elements  $\omega$  and  $\gamma$ , without loss of optimality.

Define  $\alpha = g_i^{(m)}(\omega)$  and  $\beta = g_i^{(m)}(\gamma)$ . Then  $\alpha > \beta$ . Define two new functions:

$$\begin{aligned} g_i^{(m), low}(\omega_i) &= \begin{cases} g_i^{(m)}(\omega_i) & \text{if } \omega_i \notin \{\omega, \gamma\} \\ \beta & \text{if } \omega_i \in \{\omega, \gamma\} \end{cases} \\ g_i^{(m), high}(\omega_i) &= \begin{cases} g_i^{(m)}(\omega_i) & \text{if } \omega_i \notin \{\omega, \gamma\} \\ \alpha & \text{if } \omega_i \in \{\omega, \gamma\} \end{cases} \end{aligned}$$

Unlike the original function  $g_i^{(m)}(\omega_i)$ , these new functions satisfy:

$$\begin{aligned} g_i^{(m),low}(\omega) &\leq g_i^{(m),low}(\gamma) \\ g_i^{(m),high}(\omega) &\leq g_i^{(m),high}(\gamma) \end{aligned}$$

Define  $\mathbf{g}^{(m),low}(\boldsymbol{\omega})$  and  $\mathbf{g}^{(m),high}(\boldsymbol{\omega})$  by replacing the  $i$ th component function  $g_i^{(m)}(\omega_i)$  of  $\mathbf{g}^{(m)}(\boldsymbol{\omega})$  with new component functions  $g_i^{(m),low}(\omega_i)$  and  $g_i^{(m),high}(\omega_i)$ , respectively. Let  $p_k^{old}(t)$  be the  $k$ th penalty incurred in the (old) strategy that uses  $\mathbf{g}^{(m)}(\boldsymbol{\omega})$  with probability  $\theta_m$ . Let  $p_k^{new}(t)$  be the corresponding penalty under a (new) strategy that, instead of using  $\mathbf{g}^{(m)}(\boldsymbol{\omega})$  with probability  $\theta_m$ , uses:

- $\mathbf{g}^{(m),low}(\boldsymbol{\omega})$  with probability  $\theta_m q_i(\gamma)/(q_i(\omega) + q_i(\gamma))$ .
- $\mathbf{g}^{(m),high}(\boldsymbol{\omega})$  with probability  $\theta_m q_i(\omega)/(q_i(\omega) + q_i(\gamma))$ .

Let  $\boldsymbol{\omega}_{\bar{i}}(t)$  denote the  $(N-1)$ -dimensional vector of components  $\omega_j(t)$  for  $j \neq i$ . Fix any vector  $\boldsymbol{\omega}_{\bar{i}} \in \Omega_{\bar{i}}$ . Define  $\boldsymbol{\alpha}_{\bar{i}}$  as the corresponding  $(N-1)$ -dimensional vector of  $g_j^{(m)}(\omega_j)$  values for  $j \neq i$ . Then:

- If  $\boldsymbol{\omega}_{\bar{i}}(t) = \boldsymbol{\omega}_{\bar{i}}$ ,  $\omega_i(t) = \omega$ , and  $\mathbf{g}^{(m),low}(\boldsymbol{\omega})$  is used by the new strategy, then  $\boldsymbol{\omega}(t) = [\boldsymbol{\omega}_{\bar{i}}, \omega]$  and:

$$\begin{aligned} p_k^{new}(t) &= \hat{p}_k(\mathbf{g}^{(m),low}([\boldsymbol{\omega}_{\bar{i}}, \omega]), [\boldsymbol{\omega}_{\bar{i}}, \omega]) \\ &= \hat{p}_k([\boldsymbol{\alpha}_{\bar{i}}, \beta], [\boldsymbol{\omega}_{\bar{i}}, \omega]) \end{aligned}$$

Further, since the old strategy used  $g_i^{(m)}(\omega) = \alpha$ :

$$\begin{aligned} p_k^{old}(t) &= \hat{p}_k(\mathbf{g}^{(m)}([\boldsymbol{\omega}_{\bar{i}}, \omega]), [\boldsymbol{\omega}_{\bar{i}}, \omega]) \\ &= \hat{p}_k([\boldsymbol{\alpha}_{\bar{i}}, \alpha], [\boldsymbol{\omega}_{\bar{i}}, \omega]) \end{aligned}$$

- If  $\boldsymbol{\omega}_{\bar{i}}(t) = \boldsymbol{\omega}_{\bar{i}}$ ,  $\omega_i(t) = \gamma$ , and  $\mathbf{g}^{(m),high}(\boldsymbol{\omega})$  is used by the new strategy, then  $\boldsymbol{\omega}(t) = [\boldsymbol{\omega}_{\bar{i}}, \gamma]$  and:

$$\begin{aligned} p_k^{new}(t) &= \hat{p}_k(\mathbf{g}^{(m),high}([\boldsymbol{\omega}_{\bar{i}}, \gamma]), [\boldsymbol{\omega}_{\bar{i}}, \gamma]) \\ &= \hat{p}_k([\boldsymbol{\alpha}_{\bar{i}}, \alpha], [\boldsymbol{\omega}_{\bar{i}}, \gamma]) \end{aligned}$$

Further, since the old strategy used  $g_i^{(m)}(\gamma) = \beta$ :

$$\begin{aligned} p_k^{old}(t) &= \hat{p}_k(\mathbf{g}^{(m)}([\boldsymbol{\omega}_{\bar{i}}, \gamma]), [\boldsymbol{\omega}_{\bar{i}}, \gamma]) \\ &= \hat{p}_k([\boldsymbol{\alpha}_{\bar{i}}, \beta], [\boldsymbol{\omega}_{\bar{i}}, \gamma]) \end{aligned}$$

- Suppose  $\boldsymbol{\omega}_{\bar{i}}(t) = \boldsymbol{\omega}_{\bar{i}}$ , but neither of the above two events are satisfied on slot  $t$ . That is, neither of the events  $\mathcal{E}_1$  or  $\mathcal{E}_2$  are true, where:

$$\begin{aligned} \mathcal{E}_1 &\triangleq \{\omega_i(t) = \omega\} \cap \{\mathbf{g}^{(m),low}(\boldsymbol{\omega}) \text{ is used}\} \\ \mathcal{E}_2 &\triangleq \{\omega_i(t) = \gamma\} \cap \{\mathbf{g}^{(m),high}(\boldsymbol{\omega}) \text{ is used}\} \end{aligned}$$

Then  $p_k^{new}(t) - p_k^{old}(t) = 0$ .

It follows that:

$$\begin{aligned} &\mathbb{E}[p_k^{new}(t) - p_k^{old}(t) | \boldsymbol{\omega}_{\bar{i}}(t) = \boldsymbol{\omega}_{\bar{i}}] \\ &= \theta_m q_i(\omega) \left( \frac{q_i(\gamma)}{q_i(\omega) + q_i(\gamma)} \right) \times \\ &\quad [\hat{p}_k([\boldsymbol{\alpha}_{\bar{i}}, \beta], [\boldsymbol{\omega}_{\bar{i}}, \omega]) - \hat{p}_k([\boldsymbol{\alpha}_{\bar{i}}, \alpha], [\boldsymbol{\omega}_{\bar{i}}, \omega])] \\ &\quad + \theta_m q_i(\gamma) \left( \frac{q_i(\omega)}{q_i(\omega) + q_i(\gamma)} \right) \times \\ &\quad [\hat{p}_k([\boldsymbol{\alpha}_{\bar{i}}, \alpha], [\boldsymbol{\omega}_{\bar{i}}, \gamma]) - \hat{p}_k([\boldsymbol{\alpha}_{\bar{i}}, \beta], [\boldsymbol{\omega}_{\bar{i}}, \gamma])] \quad (23) \end{aligned}$$

where the above uses the fact that  $\omega_i(t)$  is independent of  $\boldsymbol{\omega}_{\bar{i}}(t)$ , so conditioning on  $\boldsymbol{\omega}_{\bar{i}}(t) = \boldsymbol{\omega}_{\bar{i}}$  does not change the distribution of  $\omega_i(t)$ . Because  $\hat{p}_k(\cdot)$  satisfies the preferred action property and  $\alpha > \beta$ ,  $\omega < \gamma$ , one has:

$$\begin{aligned} &[\hat{p}_k([\boldsymbol{\alpha}_{\bar{i}}, \alpha], [\boldsymbol{\omega}_{\bar{i}}, \omega]) - \hat{p}_k([\boldsymbol{\alpha}_{\bar{i}}, \beta], [\boldsymbol{\omega}_{\bar{i}}, \omega])] \\ &\geq [\hat{p}_k([\boldsymbol{\alpha}_{\bar{i}}, \alpha], [\boldsymbol{\omega}_{\bar{i}}, \gamma]) - \hat{p}_k([\boldsymbol{\alpha}_{\bar{i}}, \beta], [\boldsymbol{\omega}_{\bar{i}}, \gamma])] \end{aligned}$$

and hence (23) is less than or equal to zero. This holds when conditioning on all possible values of  $\boldsymbol{\omega}_{\bar{i}}(t)$ , and so:

$$\mathbb{E}[p_k^{new}(t) - p_k^{old}(t)] \leq 0$$

This holds for all penalties  $k \in \{0, 1, \dots, K\}$ , and so the modified algorithm still satisfies all constraints with an optimal value for  $\mathbb{E}[p_0(t)]$ . The interchange can be repeated a finite number of times until all strategy functions are non-decreasing. ■

In the special case of binary actions, so that  $\mathcal{A}_i = \{0, 1\}$  for all  $i \in \{1, \dots, N\}$ , all non-decreasing strategy functions  $g_i(\omega_i)$  have the following form:

$$g_i(\omega_i) = \begin{cases} 0 & \text{if } \omega_i < h_i^* \\ 1 & \text{if } \omega_i \geq h_i^* \end{cases} \quad (24)$$

for some threshold  $h_i^* \in \{0, 1, \dots, |\Omega_i|\}$ . There are  $|\Omega_i| + 1$  such threshold functions, whereas the total number of strategy functions for user  $i$  is  $2^{|\Omega_i|}$ . Restricting to the threshold functions significantly decreases complexity.

## V. ONLINE OPTIMIZATION

This section presents a dynamic algorithm to solve the problem (12)-(14). The algorithm can also be viewed as an online solution to the linear program (16)-(19). Let  $\bar{M}$  be the number of pure strategies required for consideration in the linear program (where  $\bar{M}$  is possibly smaller than  $M$ , as discussed in the previous section). Reorder the functions  $g^{(m)}(\boldsymbol{\omega})$  if necessary so that every slot  $t$ , the system chooses a strategy function in the set  $\{g^{(1)}(\boldsymbol{\omega}), \dots, g^{(\bar{M})}(\boldsymbol{\omega})\}$ .

Suppose all users receive feedback specifying the values of the penalties  $p_1(t), \dots, p_K(t)$  at the end of slot  $t + D$ , where  $D$  is a non-negative integer that represents a system delay. For each constraint  $k \in \{1, \dots, K\}$ , define a *virtual queue*  $Q_k(t)$  and initialize  $Q_k(0)$  to a commonly known value (typically 0). For each  $t \in \{0, 1, 2, \dots\}$  the queue is updated by:

$$Q_k(t+1) = \max[Q_k(t) + p_k(t-D) - c_k, 0] \quad (25)$$

Each user can iterate the above equation based on information available at the end of slot  $t$ . Thus, all users know the value of  $Q_k(t)$  at the beginning of each slot  $t$ . If  $D > 0$ , define  $p_k(-1) = p_k(-2) = \dots = p_k(-D) = 0$ .

**Lemma 5:** Under any decision rule for choosing strategy functions over time, for all  $t > 0$  one has:

$$\frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}[p_k(\tau - D)] \leq c_k + \frac{\mathbb{E}[Q_k(t)]}{t} - \frac{\mathbb{E}[Q_k(0)]}{t}$$

**Proof:** From (25) the following holds for all slots  $\tau \in \{0, 1, 2, \dots\}$ :

$$Q_k(\tau+1) \geq Q_k(\tau) + p_k(\tau - D) - c_k$$

Thus:

$$Q_k(\tau + 1) - Q_k(\tau) \geq p_k(\tau - D) - c_k$$

Summing over  $\tau \in \{0, 1, \dots, t-1\}$  for  $t > 0$  gives:

$$Q_k(t) - Q_k(0) \geq \sum_{\tau=0}^{t-1} p_k(\tau - D) - c_k t$$

Rearranging terms proves the result.  $\blacksquare$

Lemma 5 ensures the constraints (13) are satisfied whenever the condition  $\lim_{t \rightarrow 0} \mathbb{E}[Q_k(t)]/t = 0$  holds for all  $k \in \{1, \dots, K\}$ , a condition called *mean rate stability* [1].

#### A. Lyapunov optimization

Define  $\mathbf{Q}(t) = (Q_1(t), \dots, Q_K(t))$ . Define  $L(t)$  as the squared norm of  $\mathbf{Q}(t)$  (divided by 2 for convenience later):

$$L(t) \triangleq \frac{1}{2} \|\mathbf{Q}(t)\|^2 = \frac{1}{2} \sum_{k=1}^K Q_k(t)^2$$

Define  $\Delta(t) \triangleq L(t+1) - L(t)$ , called the *Lyapunov drift*. Consider the following structure for the control decisions: Every slot  $t$  the queues  $\mathbf{Q}(t)$  are observed. Then a collection of non-negative values  $\beta_m(t)$  are created that satisfy  $\sum_{m=1}^{\tilde{M}} \beta_m(t) = 1$  (if desired, the  $\beta_m(t)$  values can be chosen as a function of the  $\mathbf{Q}(t)$  values). Then an index  $m \in \{1, \dots, \tilde{M}\}$  is randomly and independently chosen according to the probability mass function  $\beta_m(t)$ , and the decision rule  $\mathbf{g}^{(m)}(\omega(t))$  is used for slot  $t$ . Thus, a specific algorithm with this structure is determined by specifying how the  $\beta_m(t)$  probabilities are chosen on each slot  $t$ .

Motivated by the theory in [1], the approach is to choose probabilities every slot to greedily minimize a bound on the *drift-plus-penalty* expression  $\mathbb{E}[\Delta(t+D) + V p_0(t) | \mathbf{Q}(t)]$ , where  $V$  is a non-negative weight that affects a performance tradeoff. The  $D$ -shifted drift term  $\Delta(t+D)$  is different from [1] and is used because of the delayed feedback structure of the queue update (25). The intuition is that minimizing  $\Delta(t+D)$  maintains queue stability, while adding the weighted penalty term  $V p_0(t)$  biases decisions in favor of lower penalties. The following lemma provides a bound on the drift-plus-penalty expression under any  $\beta_m(t)$  probabilities.

**Lemma 6:** Fix  $V \geq 0$ . Under the above decision structure, one has for slot  $t$ :

$$\mathbb{E}[\Delta(t+D) + V p_0(t) | \mathbf{Q}(t)] \leq B(1 + 2D) + V \sum_{m=1}^{\tilde{M}} \beta_m(t) r_0^{(m)} + \sum_{k=1}^K Q_k(t) \left[ \sum_{m=1}^{\tilde{M}} \beta_m(t) r_k^{(m)} - c_k \right] \quad (26)$$

where  $r_k^{(m)}$  is the  $k$ th component of  $\mathbf{r}^{(m)}$  as defined in (15), and the constant  $B$  is defined:

$$B \triangleq \max_{m \in \{1, \dots, \tilde{M}\}} \frac{1}{2} \sum_{k=1}^K \sum_{\omega \in \Omega} \pi(\omega) \left| \hat{p}_k(\mathbf{g}^{(m)}(\omega), \omega) - c_k \right|^2$$

*Proof:* Note that for all  $k \in \{0, 1, \dots, K\}$ :

$$\begin{aligned} \mathbb{E}[p_k(t) | \mathbf{Q}(t)] &= \mathbb{E}[\hat{p}_k(\boldsymbol{\alpha}(t), \omega(t)) | \mathbf{Q}(t)] \\ &= \sum_{m=1}^{\tilde{M}} \sum_{\omega \in \Omega} \beta_m(t) \pi(\omega) \hat{p}_k(\mathbf{g}^{(m)}(\omega), \omega) \\ &= \sum_{m=1}^{\tilde{M}} \beta_m(t) r_k^{(m)} \end{aligned}$$

Therefore, to prove (26) it suffices to prove:

$$\begin{aligned} \mathbb{E}[\Delta(t+D) | \mathbf{Q}(t)] &\leq B(1 + 2D) \\ &+ \sum_{k=1}^K Q_k(t) \mathbb{E}[p_k(t) - c_k | \mathbf{Q}(t)] \end{aligned} \quad (27)$$

To this end, squaring the queue equation (25), using  $\max[a, 0]^2 \leq a^2$ , and evaluating at time  $t+D$  yields:

$$\begin{aligned} Q_k(t+D+1)^2 &\leq Q_k(t+D)^2 + (p_k(t) - c_k)^2 \\ &+ 2Q_k(t+D)(p_k(t) - c_k) \end{aligned}$$

Summing over  $k \in \{1, \dots, K\}$  and dividing by 2 gives:

$$\begin{aligned} \Delta(t+D) &\leq \frac{1}{2} \sum_{k=1}^K (p_k(t) - c_k)^2 \\ &+ \sum_{k=1}^K Q_k(t+D)(p_k(t) - c_k) \\ &= \frac{1}{2} \sum_{k=1}^K (p_k(t) - c_k)^2 \\ &+ \sum_{k=1}^K Q_k(t)(p_k(t) - c_k) \\ &+ \sum_{k=1}^K (Q_k(t+D) - Q_k(t))(p_k(t) - c_k) \end{aligned}$$

Taking conditional expectations of the above proves (27) upon application of the following inequalities (see Appendix E):

$$\begin{aligned} \frac{1}{2} \sum_{k=1}^K \mathbb{E}[(p_k(t) - c_k)^2 | \mathbf{Q}(t)] &\leq B \\ \sum_{k=1}^K \mathbb{E}[(Q_k(t+D) - Q_k(t))(p_k(t) - c_k) | \mathbf{Q}(t)] &\leq 2BD \end{aligned}$$

#### B. The drift-plus-penalty algorithm

Observe that the probability mass function  $\beta_m(t)$  that minimizes the right-hand-side of (26) is the one that, with probability 1, chooses the index  $m \in \{1, \dots, \tilde{M}\}$  that minimizes the expression (breaking ties arbitrarily):

$$V r_0^{(m)} + \sum_{k=1}^K Q_k(t) r_k^{(m)} \quad (28)$$

This gives rise to the following *drift-plus-penalty algorithm*: Every slot  $t$ :



- Users observe the queue vector  $\mathbf{Q}(t)$ .
- Users select the pure decision strategy  $\mathbf{g}^{(m)}(\omega)$ , where  $m$  is the index that minimizes the expression (28).
- The delayed penalty information  $p_k(t - D)$  is observed and queues are updated via (25).

### C. Performance Analysis

*Theorem 3:* If the problem (12)-(14) is feasible, then under the drift-plus-penalty algorithm for any  $V \geq 0$ :

- All desired constraints (13)-(14) are satisfied.
- For all  $t > 0$ , the time average expectation of  $p_0(t)$  satisfies:

$$\frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}[p_0(\tau)] \leq p_0^{opt} + \frac{B(1+2D)}{V} + \frac{\mathbb{E}[L(D)]}{Vt} \quad (29)$$

- For all  $t > 0$ , the time average expectation of  $p_k(t)$  satisfies the following for all  $k \in \{1, \dots, K\}$ :

$$\frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}[p_k(\tau)] \leq c_k + O(\sqrt{V/t}) \quad (30)$$

The above theorem shows the time average expectation of  $p_0(t)$  is within  $O(1/V)$  of optimality. It can be pushed as close to optimal as desired by increasing the  $V$  parameter. The tradeoff is in the amount of time required for the time average expected penalties to be close to their desired constraints. It can be shown that if  $D = 0$  and a mild *Slater condition* is satisfied, then the bound (30) can be improved to (see Appendix D):

$$\frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}[p_k(\tau)] \leq c_k + O(V/t) + O(\log(t)/t) \quad (31)$$

*Proof:* (Theorem 3) Every slot  $\tau \in \{0, 1, 2, \dots\}$  the drift-plus-penalty algorithm chooses probabilities  $\beta_m(\tau)$  that minimize the right-hand-side of the expression (26). Thus:

$$\mathbb{E}[\Delta(\tau + D) + Vp_0(\tau) | \mathbf{Q}(\tau)] \leq B(1+2D) \\ V \sum_{m=1}^{\tilde{M}} \theta_m r_0^{(m)} + \sum_{k=1}^K Q_k(\tau) \left[ \sum_{m=1}^{\tilde{M}} \theta_m r_k^{(m)} - c_k \right]$$

where  $\theta_m$  is any alternative probability mass function defined over  $m \in \{1, \dots, \tilde{M}\}$ . Using the probabilities  $\theta_m$  that optimally solve the linear program (16)-(19) gives:

$$\mathbb{E}[\Delta(\tau + D) + Vp_0(\tau) | \mathbf{Q}(\tau)] \leq B(1+2D) + Vp_0^{opt}$$

Taking expectations of both sides and using iterated expectations gives:

$$\mathbb{E}[\Delta(\tau + D)] + V\mathbb{E}[p_0(\tau)] \leq B(1+2D) + Vp_0^{opt}$$

Summing over  $\tau \in \{0, 1, \dots, t-1\}$  gives:

$$\mathbb{E}[L(t+D)] - \mathbb{E}[L(D)] + V \sum_{\tau=0}^{t-1} \mathbb{E}[p_0(\tau)] \leq \\ B(1+2D)t + Vp_0^{opt}t \quad (32)$$

Using the fact that  $\mathbb{E}[L(t+D)] \geq 0$  and rearranging terms proves (29).

Again rearranging (32) yields:

$$\mathbb{E}[L(t+D)] \leq (C + FV)t \quad (33)$$

where  $C$  is defined:

$$C \triangleq \mathbb{E}[L(D)] + B(1+2D)$$

and  $F$  is defined as a constant that satisfies the following for all slots  $\tau$ :

$$F \geq p_0^{opt} - \mathbb{E}[p_0(\tau)]$$

Such a constant exists because  $p_0(\tau)$  has a finite number of possible outcomes. Using the definition of  $L(t+D)$  in (33) gives:

$$\mathbb{E}[\|\mathbf{Q}(t+D)\|^2] \leq 2(C + FV)t$$

By Jensen's inequality:

$$\mathbb{E}[\|\mathbf{Q}(t+D)\|] \leq \sqrt{2(C + FV)t}$$

Thus:

$$\frac{\mathbb{E}[\|\mathbf{Q}(t+D)\|]}{t} \leq \sqrt{\frac{2(C + FV)}{t}}$$

Using this with Lemma 5 proves (30). The inequality (30) immediately implies that all desired constraints are satisfied. ■

### D. The approximate drift-plus-penalty algorithm

The algorithm of Section V-B assumes perfect knowledge of the  $r_k^{(m)}$  values. These can be computed by (15) if the event probabilities  $\pi(\omega)$  are known. Suppose these probabilities are unknown, but delayed samples  $\omega(t-D)$  are available at the end of each slot  $t$ . Let  $W$  be a positive integer that represents a *sample size*. The  $r_k^{(m)}$  values can be approximated by:

$$\tilde{r}_k^{(m)}(t) = \frac{1}{W} \sum_{w=0}^{W-1} \hat{p}_k \left( \mathbf{g}^{(m)}(\omega(t-D-w)), \omega(t-D-w) \right)$$

The approximate algorithm uses  $\tilde{r}_k^{(m)}(t)$  values in replace of  $r_k^{(m)}$  in the expression (28). Analysis in [20] shows that the performance gap between exact and approximate drift-plus-penalty implementations is  $O(1/\sqrt{W})$ , so that the approximate algorithm is very close to the exact algorithm when  $W$  is large.

### E. Separable penalty functions

A simpler and exact implementation is possible, without requiring knowledge of the probability distribution for  $\omega(t)$ , when penalty functions have the following separable form for all  $k \in \{0, 1, \dots, K\}$ :

$$\hat{p}_k(\alpha, \omega) = \sum_{i=1}^N \hat{p}_{ik}(\alpha_i, \omega_i) \quad (34)$$

where  $\hat{p}_{ik}(\alpha_i, \omega_i)$  are any functions of  $(\alpha_i, \omega_i) \in \mathcal{A}_i \times \Omega_i$ . Choosing an  $m \in \{1, \dots, \tilde{M}\}$  that minimizes the expression (28) is equivalent to observing the queues  $\mathbf{Q}(t)$  and then

choosing a strategy function  $\mathbf{g}(\boldsymbol{\omega}) = (g_1(\omega_1), \dots, g_N(\omega_N))$  to minimize:

$$\sum_{\boldsymbol{\omega} \in \Omega} \pi(\boldsymbol{\omega}) \left[ V \hat{p}_0(\mathbf{g}(\boldsymbol{\omega}), \boldsymbol{\omega}) + \sum_{k=1}^K Q_k(t) \hat{p}_k(\mathbf{g}(\boldsymbol{\omega}), \boldsymbol{\omega}) \right]$$

With the structure (34), this expression becomes:

$$\sum_{\boldsymbol{\omega} \in \Omega} \sum_{i=1}^K \pi(\boldsymbol{\omega}) \left[ V \hat{p}_{i0}(g_i(\omega_i), \omega_i) + \sum_{k=1}^K Q_k(t) \hat{p}_{ik}(g_i(\omega_i), \omega_i) \right]$$

The above is minimized by the following for each  $i \in \{1, \dots, N\}$ :

$$g_i(\omega_i) = \arg \min_{\alpha_i \in \mathcal{A}_i} \left[ V \hat{p}_{i0}(\alpha_i, \omega_i) + \sum_{k=1}^K Q_k(t) \hat{p}_{ik}(\alpha_i, \omega_i) \right]$$

Thus, the minimization step in the drift-plus-penalty algorithm reduces to having each user observe its own  $\omega_i(t)$  value and then setting  $\alpha_i(t) = g_i(\omega_i(t))$ , where the function  $g_i(\omega_i)$  is defined above. The queue update (25) is the same as before.

In the special case  $D = 0$ , this is the same algorithm as the optimal (centralized) drift-plus-penalty algorithm of [1]. Hence, for separable problems, there is no optimality gap between centralized and distributed algorithms.

## VI. SIMULATIONS

### A. Ergodic performance for a 2 user system

This subsection presents simulation results for the 2 user sensor network example of Section II. The approximate drift-plus-penalty algorithm of Section V-D is used with a delay of  $D = 10$  slots and a moving average window size of  $W = 40$  slots. The algorithm is not aware of the system probabilities. The objective of this simulation is to find how close the achieved utility is to the optimal value  $u^{opt} = 23/48 \approx 0.47917$  computed in Section II-B. Recall that the desired power constraints are  $\bar{p}_i \leq 1/3$  for each user  $i \in \{1, 2\}$ . The table in Fig. 1 presents performance for various values of  $V$ . For  $V \geq 50$  the achieved utility differs from optimality only in the fourth decimal place.

$V$	$\bar{u}$	$\bar{p}_1$	$\bar{p}_2$
1	0.344639	0.259764	0.219525
5	0.454557	0.333158	0.267161
10	0.472763	0.333335	0.300415
25	0.478186	0.333346	0.326948
50	0.479032	0.333369	0.332873
100	0.479218	0.333406	0.333334

Fig. 1. Algorithm performance over  $t = 10^6$  slots ( $D = 10$ ,  $W = 40$ ). Recall that  $u^{opt} = 23/48 \approx 0.47917$ .

### B. Ergodic performance for a 3 user system

Consider a network of 3 sensors that communicate reports to a fusion center, similar to the example considered in Section II. The event processes  $\omega_i(t)$  for each sensor  $i \in \{1, 2, 3\}$  take values in the same 10 element set  $\Omega$ :

$$\Omega \triangleq \{0, 1, 2, 3, \dots, 9\}$$

Consider binary actions  $\alpha_i(t) \in \{0, 1\}$ , where  $\alpha_i(t) = 1$  corresponds to sensor  $i$  sending a report, and incurs a power cost of 1 for that sensor. The penalty and utility functions are:

$$\begin{aligned} \hat{p}_i(\alpha_i, \omega_i) &= \alpha_i \quad \forall i \in \{1, 2, 3\} \\ \hat{u}(\boldsymbol{\alpha}, \boldsymbol{\omega}) &= \min \left[ \frac{\alpha_1 \omega_1}{10} + \frac{\alpha_2 \omega_2 + \alpha_3 \omega_3}{20}, 1 \right] \end{aligned}$$

Thus, sensor 1 brings more utility than the other sensors.

Assume  $\omega_1(t), \omega_2(t), \omega_3(t)$  are mutually independent and uniformly distributed over  $\Omega$ . The requirements for Theorem 2 hold, and so one can restrict attention to the 11 threshold functions  $g_i(\omega_i)$  of the type (24). As it does not make sense to report when  $\omega_i(t) = 0$ , the functions  $g_i(\omega) = 1$  for all  $\omega$  can be removed. This leaves only 10 threshold functions at each user, for a total of  $10^3 = 1000$  strategy functions  $\mathbf{g}^{(m)}(\boldsymbol{\omega})$  to be considered every slot. The approximate drift-plus-penalty algorithm of Section V-D is simulated over  $t = 10^6$  slots with a delay  $D = 10$  and for various choices of the moving average window size  $W$  and the parameter  $V$ . All average power constraints were met for all choices of  $V$  and  $W$ . The achieved utility is shown in Fig. 2. The utility increases to a limiting value as  $V$  is increased. This limiting value can be improved by adjusting the number of samples  $W$  used in the moving average. Increasing  $W$  from 40 to 200 gives a small improvement in performance. There is only a negligible improvement when  $W$  is further increased to 400 (the curves for  $W = 200$  and  $W = 400$  look identical).

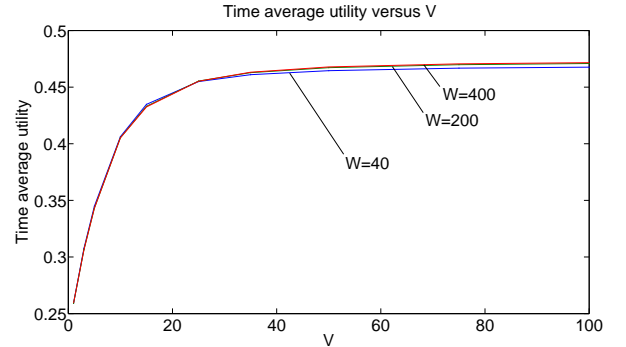


Fig. 2. Achieved utility  $\bar{u}$  versus  $V$  for various choices of  $W$ .

Fig. 4 demonstrates how the  $V$  parameter affects the rate of convergence to the desired constraints. The window size is fixed to  $W = 40$  and the value  $\max[\bar{p}_1(t), \bar{p}_2(t), \bar{p}_3(t)]$  is plotted for  $t \in \{0, 1, \dots, 2000\}$  (where  $\bar{p}_i(t)$  is the empirical average power expenditure of user  $i$  up to slot  $t$ ). This value approaches the desired constraint of  $1/3$  more slowly when  $V$  is large. The following table presents time averages after a longer duration of  $10^6$  slots.

$V$	$\bar{u}$	$\bar{p}_1$	$\bar{p}_2$	$\bar{p}_3$
1	0.259400	0.258000	0.251310	0.251342
10	0.406263	0.333301	0.316371	0.316418
50	0.464545	0.333357	0.333341	0.333342
100	0.467642	0.333387	0.333354	0.333354

Fig. 3. Time averages after  $t = 10^6$  slots ( $W = 40$ ).

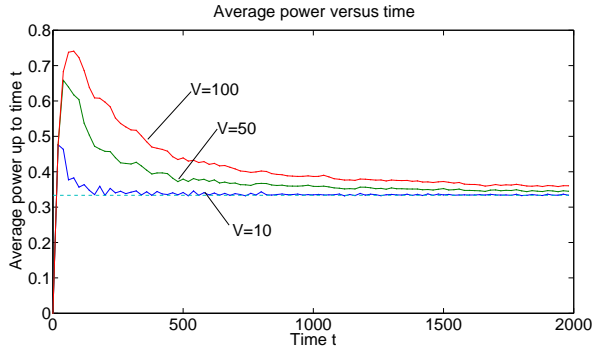


Fig. 4. An illustration of the rate of convergence to the desired constraint  $1/3$  for various choices of  $V$ . The curves plot  $\max[\bar{p}_1(t), \bar{p}_2(t), \bar{p}_3(t)]$  versus  $t$ .

### C. Adaptation to non-ergodic changes

The initial queue state determines the coefficient of an  $O(1/t)$  transient in the performance bounds of the system (consider the  $\mathbb{E}[L(D)]/(Vt)$  term in (29)). Thus, if system probabilities change abruptly, the system can be viewed as restarting with a different initial condition. Thus, one expects the system to react robustly to such changes.

To illustrate this, consider the same 3-user system of the previous subsection, using  $V = 50, W = 40$ . The event processes  $\omega_i(t)$  have the same probabilities as given in the previous subsection for slots  $t < 4000$  and  $t > 8000$ . Call this *distribution type 1*. However, for slots  $t \in \{4000, \dots, 8000\}$ , the  $\omega_i(t)$  processes are independently chosen with a different distribution as follows:

- $Pr[\omega_1(t) = 0] = Pr[\omega_1(t) = 9] = 1/2$ .
- $Pr[\omega_2(t) = k] = 1/4$  for  $k \in \{6, 7, 8, 9\}$ .
- $Pr[\omega_3(t) = k] = 1/4$  for  $k \in \{6, 7, 8, 9\}$ .

This is called *distribution type 2*.

Fig. 5 shows average utility and average power over the first 12000 slots. Values at each slot  $t$  are averaged over 2000 independent system runs. The two dashed horizontal lines in the top plot of the figure are long term time average utilities achieved over  $10^6$  slots under probabilities that are fixed at distribution type 1 and type 2, respectively. It is seen that the system adapts to the non-ergodic change by quickly adjusting to the new optimal average utility. The figure also plots average power of user 1 versus time, with a dashed horizontal line at the power constraint  $1/3$ . A noticeable disturbance in average power occurs at the non-ergodic changes in distribution.

It was observed that system performance is not very sensitive to inaccurate estimates of the  $r_k^{(m)}$  values (results not shown in the figures). This suggests that, for this example, the virtual queues alone are sufficient to ensure the average power constraints are met, which, together with loose estimates for  $r_k^{(m)}$ , are sufficient to provide an accurate approximation to optimality.

## VII. CONCLUSIONS

This paper treated distributed scheduling in a multi-user system where users know their own observations and actions, but not those of others. In this context, there is a fundamental performance gap between distributed and centralized

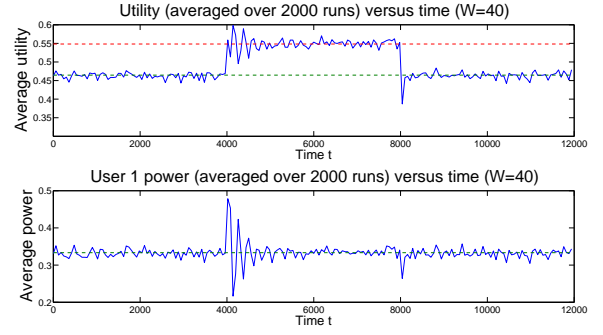


Fig. 5. A sample path of average utility and power versus time. Values at each time slot  $t$  are obtained by averaging the actual utility and power used by the algorithm on that slot over 2000 independent simulation runs.

decisions. Optimal distributed policies were constructed by correlating decisions via a source of common randomness. The optimal policy is computable via a linear program if all system probabilities are known, and through an online algorithm with virtual queues if probabilities are unknown. The online algorithm assumes there is delayed feedback about previous penalties and rewards. The algorithm was shown in simulation to adapt when system probabilities change. In the special case when the events observed at each user are independent and when penalty and utility functions satisfy a *preferred action property*, the number of pure strategies for consideration on each slot can be significantly reduced. In some cases, this reduces an exponentially complex algorithm to one that has only polynomial complexity.

## APPENDIX A — PROOF OF THEOREM 1

This appendix proves Theorem 1. Define the  $(K + 1)$ -dimensional *penalty vectors*:

$$\begin{aligned} \mathbf{p}(t) &= (p_0(t), p_1(t), \dots, p_K(t)) \\ \hat{\mathbf{p}}(\boldsymbol{\alpha}, \boldsymbol{\omega}) &= (\hat{p}_0(\boldsymbol{\alpha}, \boldsymbol{\omega}), \hat{p}_1(\boldsymbol{\alpha}, \boldsymbol{\omega}), \dots, \hat{p}_K(\boldsymbol{\alpha}, \boldsymbol{\omega})) \end{aligned}$$

For each  $m \in \{1, \dots, M\}$ , define:

$$\mathbf{r}^{(m)} \triangleq \sum_{\boldsymbol{\omega} \in \Omega} \pi(\boldsymbol{\omega}) \hat{\mathbf{p}}(\mathbf{g}^{(m)}(\boldsymbol{\omega}), \boldsymbol{\omega}) = (r_0^{(m)}, r_1^{(m)}, \dots, r_K^{(m)})$$

Define  $\mathcal{R}$  as the convex hull of these vectors:

$$\mathcal{R} \triangleq \text{Conv}(\{\mathbf{r}^{(1)}, \dots, \mathbf{r}^{(M)}\})$$

The set  $\mathcal{R}$  is convex, closed, and bounded. From the nature of the convex hull operation, the set  $\mathcal{R}$  can be viewed as the set of all average penalty vectors achievable by timesharing over the  $M$  different pure strategies.

*Lemma 7:* Let  $\boldsymbol{\alpha}(t)$  be decisions of an algorithm that satisfies the distributed scheduling constraint (11) on every slot. Then:

- (a) For all slots  $t \in \{0, 1, 2, \dots\}$ :

$$\mathbb{E}[\mathbf{p}(t)] \in \mathcal{R}$$

- (b) For all slots  $t \in \{1, 2, 3, \dots\}$ :

$$\bar{\mathbf{p}}(t) \in \mathcal{R}$$

where

$$\bar{\mathbf{p}}(t) \triangleq \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}[\mathbf{p}(\tau)]$$

*Proof:* Part (b) follows immediately from part (a) together with the fact that  $\mathcal{R}$  is convex. To prove part (a), fix a slot  $t \in \{0, 1, 2, \dots\}$ . By (11), the users make decisions:

$$\boldsymbol{\alpha}(t) = (f_1(\omega_1(t), X(t)), \dots, f_N(\omega_N(t), X(t)))$$

For each  $X(t) \in \mathcal{X}$  and  $\boldsymbol{\omega} \in \Omega$ , define:

$$\mathbf{g}_{X(t)}(\boldsymbol{\omega}) = (f_1(\omega_1, X(t)), \dots, f_N(\omega_N, X(t)))$$

Then, given  $X(t)$ , the function  $\mathbf{g}_{X(t)}(\boldsymbol{\omega})$  is a pure strategy. Hence,  $\mathbf{g}_{X(t)}(\boldsymbol{\omega}) = \mathbf{g}^{(m)}(\boldsymbol{\omega})$  for some  $m \in \{1, \dots, M\}$ . Define  $m_{X(t)}$  as the value  $m \in \{1, \dots, M\}$  for which this holds. Thus,  $\mathbf{g}_{X(t)}(\boldsymbol{\omega}) = \mathbf{g}^{(m_{X(t)})}(\boldsymbol{\omega})$ , and:

$$\begin{aligned} \mathbb{E}[\mathbf{p}(t)|X(t)] &= \mathbb{E}[\hat{\mathbf{p}}(\boldsymbol{\alpha}(t), \boldsymbol{\omega}(t))|X(t)] \\ &= \mathbb{E}\left[\hat{\mathbf{p}}\left(\mathbf{g}^{(m_{X(t)})}(\boldsymbol{\omega}(t)), \boldsymbol{\omega}(t)\right) | X(t)\right] \\ &= \sum_{\boldsymbol{\omega} \in \Omega} \pi(\boldsymbol{\omega}) \hat{\mathbf{p}}\left(\mathbf{g}^{(m_{X(t)})}(\boldsymbol{\omega}), \boldsymbol{\omega}\right) \\ &= \mathbf{r}^{(m_{X(t)})} \end{aligned}$$

Taking expectations of both sides and using the law of iterated expectations gives:

$$\mathbb{E}[\mathbf{p}(t)] = \sum_{m=1}^M \Pr[m_{X(t)} = m] \mathbf{r}^{(m)}$$

The above is a convex combination of  $\{\mathbf{r}^{(1)}, \dots, \mathbf{r}^{(M)}\}$ , and hence is in  $\mathcal{R}$ . ■

*Lemma 8:* There exist real numbers  $r_1, r_2, \dots, r_K$  that satisfy the following:

$$r_k \leq c_k \quad \forall k \in \{1, \dots, K\} \quad (35)$$

$$(p_0^{opt}, r_1, r_2, \dots, r_K) \in \mathcal{R} \quad (36)$$

Furthermore, the vector in (36) is on the *boundary* of  $\mathcal{R}$ .

*Proof:* Fix  $q$  as a positive integer. Consider an algorithm that satisfies the distributed scheduling constraint (11) every slot. For  $k \in \{0, 1, \dots, K\}$ , let  $\bar{p}_k(t)$  be the resulting time average expected penalties. Assume the algorithm satisfies:

$$p_0^{opt} \leq \limsup_{t \rightarrow \infty} \bar{p}_0(t) \leq p_0^{opt} + 1/q \quad (37)$$

$$\limsup_{t \rightarrow \infty} \bar{p}_k(t) \leq c_k \quad \forall k \in \{1, \dots, K\} \quad (38)$$

Such an algorithm must exist because  $p_0^{opt}$  is the infimum objective value for (12) over all algorithms that satisfy the constraints (13)-(14).

Lemma 7 implies that  $\bar{\mathbf{p}}(t) = (\bar{p}_0(t), \dots, \bar{p}_K(t)) \in \mathcal{R}$  for all  $t > 0$ . Let  $t_n$  be a subsequence of times over which  $\bar{p}_0(t)$  achieves its lim sup. Since  $\bar{\mathbf{p}}(t_n)$  is in the closed and bounded set  $\mathcal{R}$  for all  $t_n > 0$ , the Bolzano-Wierstrass theorem implies there is a subsequence  $\bar{\mathbf{p}}(t_{n_m})$  that converges to a point  $\mathbf{r}(q) \in \mathcal{R}$ , where  $\mathbf{r}(q) = (r_0(q), \dots, r_K(q))$ . Thus:

$$r_0(q) = \lim_{m \rightarrow \infty} \bar{p}_0(t_{n_m}) = \limsup_{t \rightarrow \infty} \bar{p}_0(t) \quad (39)$$

$$r_k(q) = \lim_{m \rightarrow \infty} \bar{p}_k(t_{n_m}) \leq \limsup_{t \rightarrow \infty} \bar{p}_k(t) \quad \forall k \in \{1, \dots, K\}$$

Using (38) in the last inequality above gives:

$$r_k(q) \leq c_k \quad \forall k \in \{1, \dots, K\} \quad (40)$$

Further, substituting (39) into (37) gives:

$$p_0^{opt} \leq r_0(q) \leq p_0^{opt} + 1/q \quad (41)$$

This holds for all positive integers  $q$ . Thus,  $\{\mathbf{r}(q)\}_{q=1}^\infty$  is an infinite sequence of vectors in  $\mathcal{R}$  such that  $\mathbf{r}(q)$  satisfies (40) and (41) for all  $q \in \{1, 2, 3, \dots\}$ . Because  $\mathcal{R}$  is closed and bounded, the sequence  $\{\mathbf{r}(q)\}_{q=1}^\infty$  has a limit point  $\mathbf{r} = (r_0, r_1, \dots, r_K) \in \mathcal{R}$  that satisfies  $r_0 = p_0^{opt}$  and  $r_k \leq c_k$  for all  $k \in \{1, \dots, K\}$ . This proves (35) and (36).

To prove that  $\mathbf{r}$  is on the *boundary* of  $\mathcal{R}$ , it suffices to note that for any  $\epsilon > 0$ :

$$(p_0^{opt} - \epsilon, r_1, \dots, r_K) \notin \mathcal{R}$$

Indeed, if this were not true, it would be possible to construct a distributed algorithm that satisfies all desired constraints and yields a time average expected value of  $p_0(t)$  equal to  $p_0^{opt} - \epsilon$ , which contradicts the definition of  $p_0^{opt}$ . ■

Because  $\mathcal{R} = \text{Conv}(\{\mathbf{r}^{(1)}, \dots, \mathbf{r}^{(M)}\})$ , Lemma 8 implies there are probabilities  $\theta_m$  that sum to 1 such that:

$$(p_0^{opt}, r_1, \dots, r_K) = \sum_{m=1}^M \theta_m \mathbf{r}^{(m)}$$

Because  $\mathcal{R}$  is a  $(K+1)$ -dimensional set, Caratheodory's theorem ensures the above can be written using at most  $K+2$  non-zero  $\theta_m$  values. However, because the above vector is on the *boundary* of  $\mathcal{R}$ , a simple extension of Caratheodory's theorem ensures it can be written using at most  $K+1$  non-zero  $\theta_m$  values.<sup>2</sup> This proves Theorem 1.

## APPENDIX B — A COUNTEREXAMPLE

This appendix shows it is possible for an algorithm to satisfy the conditional independence assumption (10) while yielding expected utility strictly larger than that of any distributed algorithm. Consider a two user system with  $\omega_1(t), \omega_2(t)$  independent and i.i.d. Bernoulli processes with:

$$\Pr[\omega_i(t) = 1] = \Pr[\omega_i(t) = 0] = 1/2 \quad \forall i \in \{1, 2\}$$

The actions are constrained to:

$$\alpha_1(t) \in \{-1, 1\} \quad , \quad \alpha_2(t) \in \{-1, 1\}$$

Define the utility function:

$$\hat{u}(\alpha_1, \alpha_2, \omega_1, \omega_2) = g(\omega_1, \omega_2) \alpha_1 \alpha_2$$

where  $g(\omega_1, \omega_2) = 1 - 2\omega_1\omega_2$ . Then  $\hat{u}(\cdot) \in \{-1, 1\}$ . Fig. 6 indicates when the utility is 1.

Consider now the following *centralized algorithm*: Every slot  $t$ , observe  $(\omega_1(t), \omega_2(t))$  and compute  $g(\omega_1(t), \omega_2(t))$ .

- If  $g(\omega_1(t), \omega_2(t)) = 1$ , independently choose:

$$(\alpha_1(t), \alpha_2(t)) = \begin{cases} (1, 1) & \text{with probability } 1/2 \\ (-1, -1) & \text{with probability } 1/2 \end{cases}$$

<sup>2</sup>This extension to points on the boundary of a convex hull can be proven using Caratheodory's theorem together with the supporting hyperplane theorem for convex sets [21].

$\omega_1$	$\omega_2$	$g(\omega_1, \omega_2)$	Conditions required for $\hat{u} = 1$
0	0	1	$\alpha_1 = \alpha_2$
0	1	1	$\alpha_1 = \alpha_2$
1	0	1	$\alpha_1 = \alpha_2$
1	1	-1	$\alpha_1 \neq \alpha_2$

Fig. 6. A table showing the conditions needed for  $\hat{u}(\alpha_1, \alpha_2, \omega_1, \omega_2) = 1$ .

- If  $g(\omega_1(t), \omega_2(t)) = -1$ , independently choose:

$$(\alpha_1(t), \alpha_2(t)) = \begin{cases} (1, -1) & \text{with probability } 1/2 \\ (-1, 1) & \text{with probability } 1/2 \end{cases}$$

The randomization ensures that regardless of  $(\omega_1(t), \omega_2(t))$ :

$$\begin{aligned} \Pr[\alpha_1(t) = 1 | \omega_1(t), \omega_2(t)] &= \frac{1}{2} \\ \Pr[\alpha_2(t) = 1 | \omega_1(t), \omega_2(t)] &= \frac{1}{2} \end{aligned}$$

and hence the conditional independence assumption (10) is satisfied. This algorithm guarantees the utility function is 1 for all possible outcomes, and so the expected utility is also 1. However, it can be shown that an optimal *distributed algorithm* is the pure strategy  $\alpha_1(t) = \alpha_2(t) = 1$  for all  $t$  (regardless of  $\omega_1(t), \omega_2(t)$ ), which yields an expected utility of only 1/2.

#### APPENDIX C — PREFERRED ACTION LEMMAS

This appendix provides proofs of Lemmas 1-4. The proofs of Lemmas 1 and 2 follow from the following lemma.

**Lemma 9:** A penalty function  $\hat{p}(\alpha, \omega)$  has the preferred action property if it satisfies the following three properties:

- $\mathcal{A}_i = \{0, 1\}$  for  $i \in \{1, \dots, N\}$ .
- $\hat{p}(\alpha, \omega)$  is non-increasing in the vector  $\omega$ . That is, for all  $\alpha \in \mathcal{A}$  and all vectors  $\omega, \gamma \in \Omega$  that satisfy  $\omega \leq \gamma$  (with inequality taken entrywise), one has

$$\hat{p}(\alpha, \omega) \geq \hat{p}(\alpha, \gamma)$$

- Given  $\alpha_i = 0$ ,  $\hat{p}(\alpha, \omega)$  does not depend on  $\omega_i$ . That is, for all  $i \in \{1, \dots, N\}$ , all possible values of  $\alpha_{\bar{i}} \in \mathcal{A}_{\bar{i}}$ ,  $\omega_{\bar{i}} \in \Omega_{\bar{i}}$ , and all  $\omega, \gamma \in \Omega_i$ , one has:

$$\hat{p}([\alpha_{\bar{i}}, 0], [\omega_{\bar{i}}, \omega]) = \hat{p}([\alpha_{\bar{i}}, 0], [\omega_{\bar{i}}, \gamma])$$

*Proof:* Fix  $i \in \{1, \dots, N\}$ , fix  $\alpha_{\bar{i}}, \omega_{\bar{i}}$ , and fix  $\alpha, \beta \in \{0, 1\}$ ,  $\omega, \gamma \in \Omega_i$  that satisfy  $\alpha > \beta$  and  $\omega < \gamma$ . Since  $\alpha, \beta$  are binary numbers that satisfy  $\alpha > \beta$ , it must be that  $\alpha = 1, \beta = 0$ . The goal is to show:

$$\begin{aligned} &\hat{p}([\alpha_{\bar{i}}, 1], [\omega_{\bar{i}}, \omega]) - \hat{p}([\alpha_{\bar{i}}, 0], [\omega_{\bar{i}}, \omega]) \\ &\geq \hat{p}([\alpha_{\bar{i}}, 1], [\omega_{\bar{i}}, \gamma]) - \hat{p}([\alpha_{\bar{i}}, 0], [\omega_{\bar{i}}, \gamma]) \end{aligned}$$

Since the second term on the left-hand-side is the same as the second term on the right-hand-side, it suffices to show:

$$\hat{p}([\alpha_{\bar{i}}, 1], [\omega_{\bar{i}}, \omega]) \geq \hat{p}([\alpha_{\bar{i}}, 1], [\omega_{\bar{i}}, \gamma])$$

The above inequality is true because  $\omega < \gamma$  and  $\hat{p}(\alpha, \omega)$  is non-increasing in the vector  $\omega$ . ■

*Proof:* (Lemma 1) Suppose:

$$\hat{p}(\alpha, \omega) = -\min \left[ \sum_{i=1}^N \phi_i(\omega_i) \alpha_i, b \right]$$

where  $\mathcal{A}_i = \{0, 1\}$  for  $i \in \{1, \dots, N\}$ ,  $b$  is a real number, and all functions  $\phi_i(\omega_i)$  are non-decreasing in  $\omega_i$ . Then  $\hat{p}(\alpha, \omega)$  is non-increasing in the  $\omega$  vector. Furthermore, for any given  $i \in \{1, \dots, N\}$ , any  $\alpha_{\bar{i}} \in \mathcal{A}_{\bar{i}}$ ,  $\omega_{\bar{i}} \in \Omega_{\bar{i}}$ , and any  $\omega, \gamma \in \Omega_i$ , one has:

$$\begin{aligned} \hat{p}([\alpha_{\bar{i}}, 0], [\omega_{\bar{i}}, \omega]) &= -\min \left[ \sum_{j \neq i} \phi_j(\omega_j) \alpha_j, b \right] \\ &= \hat{p}([\alpha_{\bar{i}}, 0], [\omega_{\bar{i}}, \gamma]) \end{aligned}$$

Thus,  $\hat{p}(\alpha, \omega)$  satisfies the requirements of Lemma 9. ■

*Proof:* (Lemma 2) Suppose:

$$\hat{p}(\alpha, \omega) = -\sum_{i=1}^N \omega_i \alpha_i \prod_{j \neq i} (1 - \alpha_j)$$

where  $\alpha_i \in \{0, 1\}$  and  $\omega_i \in \{0, 1, \dots, |\Omega_i| - 1\}$  for all  $i \in \{1, \dots, N\}$ . Then  $\hat{p}(\alpha, \omega)$  is non-increasing in the  $\omega$  vector. Now fix  $i \in \{1, \dots, N\}$ , fix  $\alpha_{\bar{i}}, \omega_{\bar{i}}$ , and fix  $\omega, \gamma \in \Omega_i$ . Then:

$$\begin{aligned} \hat{p}([\alpha_{\bar{i}}, 0], [\omega_{\bar{i}}, \omega]) &= -\sum_{k \neq i} \omega_k \alpha_k \prod_{j \neq k} (1 - \alpha_j) \\ &= \hat{p}([\alpha_{\bar{i}}, 0], [\omega_{\bar{i}}, \gamma]) \end{aligned}$$

Thus,  $\hat{p}(\alpha, \omega)$  satisfies the requirements of Lemma 9. ■

*Proof:* (Lemma 3) Suppose:

$$\hat{p}(\alpha, \omega) = \prod_{i=1}^N \phi_i(\omega_i) \psi_i(\alpha_i)$$

where  $\phi_i(\omega_i)$  is non-negative and non-increasing in  $\omega_i$  and  $\psi_i(\alpha_i)$  is non-negative and non-decreasing in  $\alpha_i$ . Fix  $i \in \{1, \dots, N\}$ , fix  $\alpha_{\bar{i}}, \omega_{\bar{i}}$ , and fix  $\alpha, \beta \in \mathcal{A}_i$ ,  $\omega, \gamma \in \Omega_i$  that satisfy  $\alpha > \beta$  and  $\omega < \gamma$ . The goal is to show:

$$\begin{aligned} &\hat{p}([\alpha_{\bar{i}}, \alpha], [\omega_{\bar{i}}, \omega]) - \hat{p}([\alpha_{\bar{i}}, \beta], [\omega_{\bar{i}}, \omega]) \\ &\geq \hat{p}([\alpha_{\bar{i}}, \alpha], [\omega_{\bar{i}}, \gamma]) - \hat{p}([\alpha_{\bar{i}}, \beta], [\omega_{\bar{i}}, \gamma]) \end{aligned}$$

By canceling common (non-negative) factors, it suffices to show:

$$\phi_i(\omega) \psi_i(\alpha) - \phi_i(\omega) \psi_i(\beta) \geq \phi_i(\gamma) \psi_i(\alpha) - \phi_i(\gamma) \psi_i(\beta)$$

This is equivalent to:

$$\phi_i(\omega) (\psi_i(\alpha) - \psi_i(\beta)) \geq \phi_i(\gamma) (\psi_i(\alpha) - \psi_i(\beta)) \quad (42)$$

Since  $\alpha > \beta$  and  $\psi_i(\alpha)$  is non-decreasing, one has  $\psi_i(\alpha) - \psi_i(\beta) \geq 0$ . By canceling the common (non-negative) factor, it suffices to show:

$$\phi_i(\omega) \geq \phi_i(\gamma)$$

This is true because  $\omega < \gamma$  and  $\phi_i(\omega)$  is non-increasing. ■

*Proof:* (Lemma 4) Suppose:

$$\hat{p}(\alpha, \omega) = \sum_{r=1}^R w_r \hat{p}_r(\alpha, \omega)$$

where  $w_r$  are non-negative constants, and each function  $\hat{p}_r(\alpha, \omega)$  has the preferred action property. Fix  $i \in \{1, \dots, N\}$ , fix  $\alpha_{\bar{i}}, \omega_{\bar{i}}$ , and fix  $\alpha, \beta \in \mathcal{A}_i$ ,  $\omega, \gamma \in \Omega_i$  that

satisfy  $\alpha > \beta$  and  $\omega < \gamma$ . Since each function  $\hat{p}_r(\alpha, \omega)$  has the preferred action property, one has for all  $r \in \{1, \dots, R\}$ :

$$\begin{aligned} & \hat{p}_r([\alpha_i, \alpha], [\omega_i, \omega]) - \hat{p}_r([\alpha_i, \beta], [\omega_i, \omega]) \\ & \geq \hat{p}_r([\alpha_i, \alpha], [\omega_i, \gamma]) - \hat{p}_r([\alpha_i, \beta], [\omega_i, \gamma]) \end{aligned}$$

Multiplying the above inequality by  $w_r$  and summing over  $r \in \{1, \dots, R\}$  proves that  $\hat{p}(\alpha, \omega)$  has the preferred action property. ■

#### VIII. APPENDIX D — THE SLATER CONDITION

For a given real number  $\epsilon \geq 0$ , consider the following linear program that is related to the linear program (16)-(19):

$$\text{Minimize:} \quad \sum_{m=1}^M \theta_m r_0^{(m)} \quad (43)$$

$$\text{Subject to:} \quad \sum_{m=1}^M \theta_m r_k^{(m)} \leq c_k - \epsilon \quad \forall k \in \{1, \dots, K\} \quad (44)$$

$$\theta_m \geq 0 \quad \forall m \in \{1, \dots, M\} \quad (45)$$

$$\sum_{m=1}^M \theta_m = 1 \quad (46)$$

If  $\epsilon > 0$ , the penalty constraints are tighter above than in the linear program (16)-(19) (compare (44) and (17)). Define  $G(\epsilon)$  as the optimal objective value (43) as a function of the parameter  $\epsilon$ . Then  $G(0) = p_0^{opt}$ , where  $p_0^{opt}$  corresponds to the original linear program (16)-(19). Define  $\epsilon_{max}$  as the largest value of  $\epsilon$  for which (43)-(46) is feasible. Suppose  $\epsilon_{max} > 0$ . This means it is possible to satisfy the desired time average penalty constraints with a slackness of  $\epsilon_{max}$  in each constraint  $k \in \{1, \dots, K\}$ . The condition  $\epsilon_{max} > 0$  is called the *Slater condition* [22].

For simplicity of exposition, assume  $D = 0$ . Since the drift-plus-penalty algorithm takes actions that minimize the right-hand-side of (26) over all probability mass functions  $\beta_m(t)$ , one has:

$$\begin{aligned} & \mathbb{E}[\Delta(t) + V p_0(t) | \mathbf{Q}(t)] \leq B \\ & V \sum_{m=1}^{\tilde{M}} \theta_m r_0^{(m)} + \sum_{k=1}^K Q_k(t) \left[ \sum_{m=1}^{\tilde{M}} \theta_m r_k^{(m)} - c_k \right] \end{aligned}$$

for any values  $\theta_m$  that satisfy (45)-(46). Using  $\theta_m$  values that solve (43)-(46) for the case  $\epsilon = \epsilon_{max}$  gives:

$$\begin{aligned} & \mathbb{E}[\Delta(t) + V p_0(t) | \mathbf{Q}(t)] \leq B \\ & V G(\epsilon_{max}) - \epsilon_{max} \sum_{k=1}^K Q_k(t) \end{aligned}$$

Therefore, for all slots  $t \in \{0, 1, 2, \dots\}$  one has:

$$\mathbb{E}[\Delta(t) | \mathbf{Q}(t)] \leq B + FV - \epsilon_{max} \sum_{k=1}^K Q_k(t) \quad (47)$$

where  $F$  is a constant that satisfies the following for all slots  $t$  and all possible values of  $\mathbf{Q}(t)$ :

$$F \geq G(\epsilon_{max}) - \mathbb{E}[p_0(t) | \mathbf{Q}(t)]$$

Now define  $\delta_{max}$  as the largest possible change in  $\|\mathbf{Q}(t)\|$  from one slot to the next, so that regardless of the control decisions, one has:

$$\|\mathbf{Q}(t+1)\| - \|\mathbf{Q}(t)\| \leq \delta_{max} \quad \forall t \in \{0, 1, 2, \dots\} \quad (48)$$

Such a value  $\delta_{max}$  exists because all penalty functions  $\hat{p}_k(\alpha(t), \omega(t))$  are bounded.

*Lemma 10:* Let  $\delta_{max}$  be a positive value that satisfies (48). Let  $A$  be a non-negative real number, and let  $\epsilon > 0$ . Assume  $\|\mathbf{Q}(0)\| = 0$  with probability 1, and that for all slots  $t$  and all possible  $\mathbf{Q}(t)$  one has:

$$\mathbb{E}[\Delta(t) | \mathbf{Q}(t)] \leq A - \epsilon \sum_{k=1}^K Q_k(t) \quad (49)$$

Then for all slots  $t \in \{1, 2, \dots\}$ :

$$\begin{aligned} & \mathbb{E}[\|\mathbf{Q}(t)\|] \leq \\ & \max \left[ \frac{\log(2)}{r}, \max \left[ \frac{2A}{\epsilon}, \frac{\epsilon}{2} \right] + \frac{\log(2t[e^{r\delta_{max}} - 1])}{r} \right] \end{aligned}$$

where  $r$  is defined:

$$r = \frac{\epsilon}{\delta_{max}^2 + \epsilon\delta_{max}/3} \quad (50)$$

Using  $A = B + FV$  in (47) shows that the system under study satisfies the requirements of the above lemma, which proves that (31) holds. The proof of the above lemma relies heavily on drift analysis in [23] and results for exponentiated martingales in [24].

*Proof:* (Lemma 10) Suppose that:

$$\|\mathbf{Q}(t)\| \geq \max[2A/\epsilon, \epsilon/2] \quad (51)$$

By definition of  $\Delta(t)$ , one has from (49):

$$\begin{aligned} & \mathbb{E}[\|\mathbf{Q}(t+1)\|^2 | \mathbf{Q}(t)] \\ & \leq \|\mathbf{Q}(t)\|^2 + 2A - 2\epsilon \sum_{k=1}^K Q_k(t) \\ & \leq \|\mathbf{Q}(t)\|^2 + 2A - 2\epsilon \|\mathbf{Q}(t)\| \quad (52) \\ & \leq \|\mathbf{Q}(t)\|^2 - \epsilon \|\mathbf{Q}(t)\| \quad (53) \\ & \leq (\|\mathbf{Q}(t)\| - \epsilon/2)^2 \end{aligned}$$

where (52) holds because the sum of the components of a non-negative vector is greater than or equal to its norm, and (53) holds because (51) implies  $\epsilon \|\mathbf{Q}(t)\| \geq 2A$ . By Jensen's inequality:

$$\mathbb{E}[\|\mathbf{Q}(t+1)\| | \mathbf{Q}(t)]^2 \leq (\|\mathbf{Q}(t)\| - \epsilon/2)^2$$

Taking the square root of both sides and using (51) gives:

$$\mathbb{E}[\|\mathbf{Q}(t+1)\mathbf{Q}(t)\|] \leq \|\mathbf{Q}(t)\| - \epsilon/2 \quad (54)$$

Define  $C$  by:

$$C \triangleq \max[2A/\epsilon, \epsilon/2]$$

so that (54) holds whenever  $\|\mathbf{Q}(t)\| \geq C$ . Define  $\delta(t)$  by:

$$\delta(t) \triangleq \|\mathbf{Q}(t+1)\| - \|\mathbf{Q}(t)\|$$

and note that  $|\delta(t)| \leq \delta_{max}$  for all  $t$ . It follows that:

$$\mathbb{E}[\delta(t) | \mathbf{Q}(t)] \leq \begin{cases} -\epsilon/2 & \text{if } \|\mathbf{Q}(t)\| \geq C \\ \delta_{max} & \text{otherwise} \end{cases} \quad (55)$$

Define  $Y(t) = e^{r\|\mathbf{Q}(t)\|}$  for a positive value of  $r$  to be determined. Assume that  $r$  satisfies:

$$0 \leq r\delta_{max} < 3 \quad (56)$$

Then:

$$\begin{aligned}
Y(t+1) - Y(t) &= e^{r\|Q(t)\|} e^{r\delta(t)} - Y(t) \\
&= Y(t)[e^{r\delta(t)} - 1] \\
&\leq \begin{cases} Y(t)[e^{r\delta(t)} - 1] & \text{if } \|Q(t)\| \geq C \\ e^{rC}[e^{r\delta_{max}} - 1] & \text{otherwise} \end{cases}
\end{aligned}$$

Now define  $g(x)$  as the function that satisfies the following for all real numbers  $x$ :

$$e^x - 1 = x + \frac{x^2}{2}g(x) \quad (57)$$

By results in [24], the function  $g(x)$  is non-decreasing in  $x$  and satisfies:

$$g(x) \leq \frac{1}{1-x/3} \quad \forall x \in [0, 3] \quad (58)$$

It follows from (57) that:

$$\begin{aligned}
e^{r\delta(t)} - 1 &= r\delta(t) + \frac{(r\delta(t))^2}{2}g(r\delta(t)) \\
&\leq r\delta(t) + \frac{(r\delta_{max})^2}{2}g(r\delta_{max}) \\
&\leq r\delta(t) + \frac{(r\delta_{max})^2}{2(1-r\delta_{max}/3)}
\end{aligned}$$

where the final inequality uses (58), which is justified because  $r\delta_{max}$  satisfies (56). Thus:

$$\begin{aligned}
Y(t+1) - Y(t) &\leq \begin{cases} Y(t)[r\delta(t) + \frac{(r\delta_{max})^2}{2(1-r\delta_{max}/3)}] & \text{if } \|Q(t)\| \geq C \\ e^{rC}[e^{r\delta_{max}} - 1] & \text{otherwise} \end{cases}
\end{aligned}$$

Taking expectations and using (55) gives:

$$\begin{aligned}
\mathbb{E}[Y(t+1) - Y(t)|Q(t)] &\leq \begin{cases} Y(t)[\frac{-r\epsilon}{2} + \frac{(r\delta_{max})^2}{2(1-r\delta_{max}/3)}] & \text{if } \|Q(t)\| \geq C \\ e^{rC}[e^{r\delta_{max}} - 1] & \text{otherwise} \end{cases}
\end{aligned}$$

Now choose  $r$  so that:

$$\frac{r\epsilon}{2} = \frac{(r\delta_{max})^2}{2(1-r\delta_{max}/3)}$$

This holds for  $r$  as defined in (50), and this choice of  $r$  maintains the inequality (56). Thus:

$$\begin{aligned}
\mathbb{E}[Y(t+1) - Y(t)|Q(t)] &\leq \begin{cases} 0 & \text{if } \|Q(t)\| \geq C \\ e^{rC}[e^{r\delta_{max}} - 1] & \text{otherwise} \end{cases}
\end{aligned}$$

Therefore, for all slots  $t$ :

$$\mathbb{E}[Y(t+1) - Y(t)] \leq e^{rC}[e^{r\delta_{max}} - 1]$$

Summing the above over  $\tau \in \{0, 1, \dots, t-1\}$  for some integer  $t > 0$  gives:

$$\mathbb{E}[Y(t)] - \mathbb{E}[Y(0)] \leq e^{rC}[e^{r\delta_{max}} - 1]t$$

Since  $Y(0) = 1$  with probability 1, and  $Y(t) = e^{r\|Q(t)\|}$ , one has:

$$\mathbb{E}[e^{r\|Q(t)\|}] - 1 \leq e^{rC}[e^{r\delta_{max}} - 1]t$$

By Jensen's inequality for the convex function  $e^x$  one has:

$$e^{r\mathbb{E}[\|Q(t)\|]} - 1 \leq e^{rC}[e^{r\delta_{max}} - 1]t$$

Thus:

$$\begin{aligned}
r\mathbb{E}[\|Q(t)\|] &\leq \log(1 + e^{rC}[e^{r\delta_{max}} - 1]t) \\
&\leq \max[\log(2), \log(2e^{rC}[e^{r\delta_{max}} - 1]t)] \\
&\leq \max[\log(2), rC + \log(2[e^{r\delta_{max}} - 1])]
\end{aligned}$$

Dividing the above by  $r$  gives the following, which holds for all integers  $t > 0$ :

$$\mathbb{E}[\|Q(t)\|] \leq \max\left[\frac{\log(2)}{r}, C + \frac{\log(2[e^{r\delta_{max}} - 1])}{r}\right]$$

■

#### APPENDIX E — THE CONSTANT IN THEOREM 3

This appendix proves the inequality involving the  $2BD$  constant at the end of the proof of Theorem 3. From (25) one has for all queues  $k \in \{1, 2, \dots, K\}$  and all slots  $\tau$ :

$$|Q_k(\tau+1) - Q_k(\tau)| \leq |p_k(\tau - D) - c_k|$$

Thus, for all slots  $t$ :

$$\begin{aligned}
|Q_k(t+D) - Q_k(t)| &\leq \sum_{d=1}^D |Q_k(t+d) - Q_k(t+d-1)| \\
&\leq \sum_{d=1}^D |p_k(t+d-1-D) - c_k| \\
&= \sum_{d=1}^D |p_k(t_d) - c_k|
\end{aligned}$$

where for notational simplicity  $t_d$  has been defined:

$$t_d \triangleq t + d - 1 - D$$

Thus:

$$\begin{aligned}
&\sum_{k=1}^K (Q_k(t+D) - Q_k(t))(p_k(t) - c_k) \\
&\leq \sum_{k=1}^K \sum_{d=1}^D |p_k(t_d) - c_k| |p_k(t) - c_k|
\end{aligned}$$

Taking expectations of the above and using the Cauchy-Schwartz inequality:<sup>3</sup>

$$\begin{aligned}
&\mathbb{E}\left[\sum_{k=1}^K (Q_k(t+D) - Q_k(t))(p_k(t) - c_k)\right] \\
&\leq \sum_{k=1}^K \sum_{d=1}^D \sqrt{\mathbb{E}[|p_k(t_d) - c_k|^2]} \sqrt{\mathbb{E}[|p_k(t) - c_k|^2]} \\
&\leq \sum_{d=1}^D \sqrt{\sum_{k=1}^K \mathbb{E}[|p_k(t_d) - c_k|^2]} \sqrt{\sum_{k=1}^K \mathbb{E}[|p_k(t) - c_k|^2]}
\end{aligned}$$

<sup>3</sup>Strictly speaking, these expectations should be conditioned on  $Q(t)$  to match with the inequalities at the end of Theorem 3. That explicit conditioning has been suppressed to simplify the expressions.

where the final inequality follows because the inner product of two vectors is less than or equal to the product of norms. The right hand side is less than or equal to:

$$\sum_{d=1}^D \sqrt{2B} \sqrt{2B} = 2BD$$

## REFERENCES

- [1] M. J. Neely. *Stochastic Network Optimization with Application to Communication and Queueing Systems*. Morgan & Claypool, 2010.
- [2] B. Liu, P. Terlechy, A. Bar-Noy, R. Govindan, M. J. Neely, and D. Rawitz. Optimizing information credibility in social swarming applications. *IEEE Trans. on Parallel and Distributed Systems*, vol. 23, no. 6, pp. 1147-1158, June 2012.
- [3] N. Michelusi and M. Zorzi. Optimal random multiaccess in energy harvesting wireless sensor networks. *Proc. IEEE International Conference on Communications*, to appear.
- [4] M. J. Osborne and A. Rubinstein. *A Course in Game Theory*. MIT Press, Cambridge, MA, 1994.
- [5] Y. Shoham and K. Leyton-Brown. *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge University Press, NY, NY, 2009.
- [6] R. Aumann. Subjectivity and correlation in randomized strategies. *Journal of Mathematical Economics*, vol. 1, pp. 67-96, 1974.
- [7] R. Aumann. Correlated equilibrium as an expression of bayesian rationality. *Econometrica*, vol. 55, pp. 1-18, 1987.
- [8] L. Georgiadis, M. J. Neely, and L. Tassiulas. Resource allocation and cross-layer control in wireless networks. *Foundations and Trends in Networking*, vol. 1, no. 1, pp. 1-149, 2006.
- [9] M. J. Neely, E. Modiano, and C. Li. Fairness and optimal stochastic control for heterogeneous networks. *IEEE/ACM Transactions on Networking*, vol. 16, no. 2, pp. 396-409, April 2008.
- [10] X. Lin and N. B. Shroff. Joint rate control and scheduling in multihop wireless networks. *Proc. of 43rd IEEE Conf. on Decision and Control, Paradise Island, Bahamas*, Dec. 2004.
- [11] L. Xiao, M. Johansson, and S. P. Boyd. Simultaneous routing and resource allocation via dual decomposition. *IEEE Transactions on Communications*, vol. 52, no. 7, pp. 1136-1144, July 2004.
- [12] S. H. Low and D. E. Lapsley. Optimization flow control, i: Basic algorithm and convergence. *IEEE/ACM Transactions on Networking*, vol. 7 no. 6, pp. 861-875, Dec. 1999.
- [13] M. Rabbat and R. Nowak. Distributed optimization in sensor networks. *Proc. IPSN*, 2004.
- [14] M. J. Neely. Distributed and secure computation of convex programs over a network of connected processors. *DCDIS Conf., Guelph, Ontario*, July 2005.
- [15] C. C. Moallemi and B. Van Roy. Distributed optimization in adaptive networks. *Advances in Neural Information Processing Systems*, vol. 16, MIT Press, 2004.
- [16] L. Jiang and J. Walrand. A distributed csma algorithm for throughput and utility maximization in wireless networks. *Proc. Allerton Conf. on Communication, Control, and Computing*, Sept. 2008.
- [17] S. Rajagopalan and D. Shah. Reversible networks, distributed optimization, and network scheduling: What do they have in common? *Proc. Conf. on Information Sciences and Systems (CISS)*, 2008.
- [18] L. Jiang and J. Walrand. *Scheduling and Congestion Control for Wireless and Processing Networks*. Morgan & Claypool, 2010.
- [19] A. Nayyar. *Sequential Decision Making in Decentralized Systems*. PhD thesis, University of Michigan, 2011.
- [20] M. J. Neely, S. T. Rager, and T. F. La Porta. Max weight learning algorithms for scheduling in unknown environments. *IEEE Transactions on Automatic Control*, vol. 57, no. 5, pp. 1179-1191, May 2012.
- [21] D. P. Bertsekas, A. Nedic, and A. E. Ozdaglar. *Convex Analysis and Optimization*. Boston: Athena Scientific, 2003.
- [22] D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, Belmont, MA, 1995.
- [23] L. Huang and M. J. Neely. Delay reduction via Lagrange multipliers in stochastic network optimization. *IEEE Transactions on Automatic Control*, vol. 56, no. 4, pp. 842-857, April 2011.
- [24] F. Chung and L. Lu. Concentration inequalities and martingale inequalities—a survey. *Internet Mathematics*, vol. 3, pp. 79-127, 2006.